Designing m-Health Modules with Sensor Interfaces for DSP Education

by

Deepta Rajan

A Thesis Presented in Partial Fulfillment
of the Requirement for the Degree
Master of Science

Approved November 2013 by the
Graduate Supervisory Committee:

Andreas Spanias, Chair
David Frakes
Pavan Turaga

ARIZONA STATE UNIVERSITY

December 2013

ABSTRACT

Advancements in mobile technologies have significantly enhanced the capabilities of mobile devices to serve as powerful platforms for sensing, processing, and visualization. Surges in the sensing technology and the abundance of data have enabled the use of these portable devices for real-time data analysis and decision-making in digital signal processing (DSP) applications. Most of the current efforts in DSP education focus on building tools to facilitate understanding of the mathematical principles. However, there is a disconnect between real-world data processing problems and the material presented in a DSP course. Sophisticated mobile interfaces and apps can potentially play a crucial role in providing a hands-on-experience with modern DSP applications to students. In this work, a new paradigm of DSP learning is explored by building an interactive easy-to-use health monitoring application for use in DSP courses. This is motivated by the increasing commercial interest in employing mobile phones for real-time health monitoring tasks. The idea is to exploit the computational abilities of the Android platform to build m-Health modules with sensor interfaces. In particular, appropriate sensing modalities have been identified, and a suite of software functionalities have been developed. Within the existing framework of the AJDSP app, a graphical programming environment, interfaces to on-board and external sensor hardware have also been developed to acquire and process physiological data. The set of sensor signals that can be monitored include electrocardiogram (ECG), photoplethysmogram (PPG), accelerometer signal, and galvanic skin response (GSR). The proposed m-Health modules can be used to estimate parameters such as heart rate, oxygen saturation, step count, and heart rate variability. A set of laboratory exercises have been designed to demonstrate the use of these modules in DSP courses. The app was evaluated through several workshops involving graduate and undergraduate students in signal processing majors at Arizona State University. The usefulness of

the software modules in enhancing student understanding of signals, sensors and DSP systems were analyzed. Student opinions about the app and the proposed m-health modules evidenced the merits of integrating tools for mobile sensing and processing in a DSP curriculum, and familiarizing students with challenges in modern data-driven applications.

To my dearest Grandmother, an artistic genius, an inspiration...

ACKNOWLEDGEMENTS

I would like to start by acknowledging the forces that allowed me to go on this journey of the masters thesis at Arizona State University. It has been a very fulfilling, adventurous and a wonderful experience for me and I thoroughly enjoyed myself these past two years.

The thesis experience would never have happened if not for my advisor Dr. Andreas Spanias, who has been extremely kind in offering all the support a student that has come from a far away land would ever need. I would like to thank him for being patient with my slow learning patterns and for trusting me and giving me the freedom to explore the field of signal processing. His motivation and constant guidance played a crucial role in the completion of this work. I would also like to express my heartfelt gratitude to Dr. David Frakes, whose unflinching help and support in all aspects related to my school life guided me through this journey from start to finish. His enthusiasm for research in diverse applications of signal/image processing gave me perspectives that allowed me to create a realizable goal for myself. I also thank my committee member Dr. Pavan Turaga for inspiring me further in the field of machine learning and for his prompt assistance with any questions I had. My special thanks to Dr. James Middleton who was always full of ideas and had an infectious enthusiasm when it came to research in engineering education. I would like to thank him for giving me the opportunity to teach a freshmen laboratory class and to run data collection experiments for software evaluation.

In addition, I am grateful to have had the opportunity to work with the following people at ASU and I would like to thank them for supporting me in various ways. Thank you Tara Grant, Rebecca Burch, Nealie Kaplan, Karen Calwell, Grisha Coleman, Assegid Kidane, Haolin Zhu, Amy Trowbridge, Chao Wang, Julian Baez and Tina Grone. I would also like to thank the engineering staff Jenna, Cynthia, Toni,

Esther, Darleen, Merri, Matt and Trudy for their timely and kind assistance.

A large portion of this journey was shared with my fellow labmates: Jayaraman and Karthikeyan who have been a great source of inspiration. I owe all my achievements to their belief in me. I thank them for motivating me and for offering to help against all odds with every little concern I had. I especially thank them for being a patient ear to my never ending questions and for participating in research discussions. I would also like to acknowledge Mahesh, Prasanna, Suhas, Girish, Bob, Sai, Sophia, Henry, Song, Mohit, and Steven. A big thank you to all of them for their help.

Now, it's time to reach my arm out to all my dear friends and my family without whom it would have been impossible to accomplish my goals. To start with, I would like to thank my family for their unending support no matter what I choose to do. Next, I would like to mention that having met Jay, Karthi, and Harini has been a life changing experience and I would like to thank them for always being there. A few other people that I would like to thank deeply for making their presence felt in my life despite being far away are Varun, Urmila, Pramaan and Shilpa. Thank you so much for being great companions. Last but certainly not the least, I would like to thank Subhiksha, Sriranjani, Apoorva, Rushil, Ramsri, Ananth and Anup for being very supportive friends and making me feel at home in Tempe.

TABLE OF CONTENTS

CHAPTER Page

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

Figure Page

Figure Page

Chapter 1

INTRODUCTION

1.1 Mobile Sensing Technologies

Today's physical world is richly interwoven with sensors, actuators, and computational elements, embedded seamlessly in the everyday objects and connected through a network [1]. Fig. 1.1 [2, 3] is an illustration of such a physical world. In addition, the advancements in mobile technologies are enabling the use of mobile devices as a powerful platform primarily for sensing, processing, and visualization. The mere prevalence of mobile technologies demands a need to devise methods that will constructively integrate these devices into our daily lives. Moreover, there is an abundance of data all around us, and mobile devices provide a way to tap into such data instantaneously and analyze it to make effective decisions in real-time. A num-

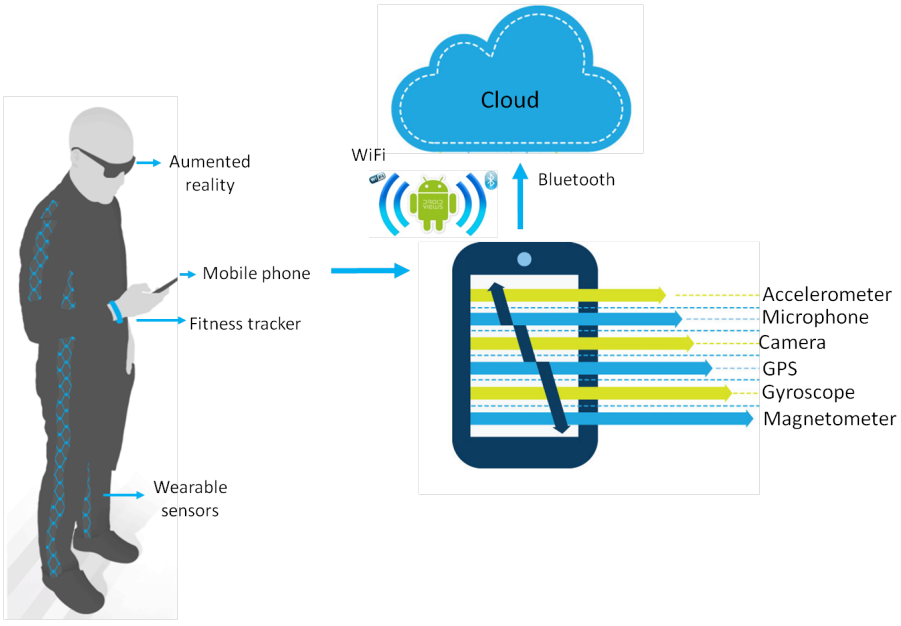


Figure 1.1: Advancements in mobile sensing technology and wearable computing.

Component	Function	Parameter	Application
Microphone	Voice	Sound	Spectrum Analysis
Camera	Picture/Video	Light	Image/Scene Analysis
Accelerometer	Rotating Screen	Acceleration	Vibrations
GPS	Position/Clock	Radio Frequency	Position
Gyroscope	Posture of devices	Angular rate	Position and Attitude

Figure 1.2: A list of Android device sensors and their applications.

ber of companies are competing to build even more such sophisticated and portable computational gadgets which further supports the use of these devices for research and education.

The current generation of mobile devices comprise sufficient memory, processing power, and a rich set of on-board peripherals such as accelerometers, digital magnetometers, gyroscopes, GPS receivers, microphones, and cameras [4]. The Android platform is one such popular mobile operating system that is expected to hold 48.8% of the mobile phone market share by 2015 [5]. To solve data-driven problems, experiencing and understanding their nature of acquisition plays an important role. Android devices provide a way for such experiences at the early stages of data sensing with its ability to capture various signals, a few of which are listed in Fig. 3 along with their application. The Android system architecture from bottom up includes the Linux Kernel, Native C/C++ libraries for various drivers, application framework such as activity managers, content providers etc., and the widgets and applications which form the top most level. It also has the Dalvik run time environment, a memory optimized java virtual machine (JVM), which compiles .class and .jar files into machine independent byte codes called .dex (dalvik executable) files.

More recently, mobile devices have also been used as a platform for scientific visualization through interfaces that allow graphical representation of complex data.

Visualization is a key component in the numerical exploration of data and it pertains to intuitive data representation, mobile visualization, augmented reality and integration of real-world geometries. Researchers from the Scientific Computing and Imaging (SCI) group at the University of Utah have developed a lightweight and highly scalable visualization environment called ViSUS [6]. It is designed to run on mobile platforms and has the ability to process massive data as it moves. Another group at the Engineering Mathematics and Computing Lab (ECML) at Hiedelberg University is investigating the impact of scientific visualization. They believe it can provide access to high performance computing [7].

1.2 Mobile Health (m-Health)

Mobile devices have created a huge impact in numerous signal processing applications with health monitoring being one of the most sought after domain. There is a vast scope for signal processing in the medical field and problems concerning biosignals and medical images comprise of a large majority of the ones being invested in and investigated. The enormous interest in healthcare has been further fueled with the advent of the handy mobile devices which have opened the door to providing easy and timely access to health-related information. Utilizing these mobile devices as instruments to monitor health has been known as m-Health, meaning mobile health.

According to a 2011 press release statement from Juniper Research, the number of mobile healthcare and medical applications (apps) downloads has reached 44 million and will reach 142 million globally by 2016. Moreover, as stated by eClinicalWorks, 89% of physicians would recommend a health app to a patient. Factors like ease-of-use, availability, and affordability of mobile devices can thus be utilized to help patients and physicians monitor health. For instance, parameters such as pulse rate, blood pressure, oxygen saturation, and skin conductance, shown to vary based

on physical activity and stress, can be measured using mobile devices and sensors. Furthermore, records of the estimated parameters can be stored for reference and communicated to doctors when required. By exploiting the pervasiveness of wireless mobile technologies and discovering ways to provide healthcare aims to revolutionize medical treatment and drastically reduce its costs in the coming years.

1.2.1 Research in m-Health

Research in mobile healthcare primarily focuses on extraction of vital physiological parameters, secure upload/transmission of data, development of sophisticated algorithms for processing, and building systems that harmoniously interface patients and physicians. Some of the application areas that are being targeted using mobile devices and wearable computing are: detecting arrhytmias [8], wheezes [9], sleep monitoring [10], stress [11], physiological parameters [12] and cardiovascular function [13]. In particular, human activity recognition opens the door to a world of healthcare applications such as elder care, cognitive assistance and fitness monitoring [14]. Multidimensional tracking of a user's well being using sensor embedded Android smartphones have been investigated in [15], where sleep patterns, physical activity and social interactions are monitored. Several front end Bluetooth enabled mobile devices have also been integrated to create a mobile care system with alert mechanisms [16]. A group of researchers from the university of Columbia and the Dartmouth college developed the CenceMe application to infer people's presence and share the information through social networks [17]. Mobile healthcare holds great promise for the future with start-ups such as Mobisante [18], FitBit [19], and Ginger.io [20] among numerous others running businesses with at least one of the applications discussed.

Figure 1.3: Depiction of the large number of apps available to users and the ability to upload information to the cloud.

1.2.2 m-Health Apps

A plethora of apps exist in the market to monitor health and provide information based on measurements acquired using on-board device sensors. Some of these apps also connect to remote databases containing patient health records, represented in Fig. 1.3 [21]. A few examples are: (a) Endomondo which monitors users engaged in activities such as running, walking, and cycling [22]; (b) Instant Heart Rate Pro is an app that measures heart rate and allows sharing information via internet; (c) The Stress Check app measures stress levels of a person using their heart rate [23]; (d) Blood Pressure Journal that measures blood pressure, heart rate, custom medication doses, weight, and body mass index (BMI) [24]; and (e) FitBit, which automatically and wirelessly records data such as steps, distances, and calories burned, obtained from a FitBit tracker. It enables daily monitoring of food, activity, weight, water, and sleep [19]. However, these apps do not have the facility to allow users to manipulate or analyze the data using digital signal processing (DSP) tools, and therefore have no scope to be used for DSP education and research. This bring us to the underlying goal of the work presented in this document which is, to combine the sensing, processing, and visualization strengths of mobile devices to create an impact in DSP

education through health monitoring applications. Some crucial aspects to consider in achieving this goal would be the choice of sensors to use, processing modules to embed, and the design of the interfaces so as to maximize the utility of the software as a learning tool in engineering education.

1.2.3 Apps in STEM Education

A large number of apps have been developed to assist teachers and students in various tasks related to coursework. Some examples are Blackboard Mobile [25], Class Dojo [26], Calculus Tools [27], and Cram [28]. Popular Android apps available for learning DSP principles include MATLAB Mobile [29], WolframAlpha [30], and Octave [31]. Several of these apps typically have a command-line interface and require Internet connectivity to interface with the cloud. In addition, MATLAB Mobile and WolframAlpha also require an active license. Although these and other applications available in the market today are compelling, most do not provide a way to design and configure simulations and visualize results. To address these drawbacks, we have developed iOS and Android graphical programming apps (iJDSP and AJDSP) that can be used to complement instruction in graduate and undergraduate DSP courses [32, 33, 34]. The iJDSP app is available for free download on the iTunes App Store [35], while the AJDSP app (Fig. 1.4) is available for free download on the Google Play Store [36]. Each DSP functionality is developed as a 'block', and they can be broadly categorized into signal acquisition modules and signal processing modules.

1.3 Mobile Devices in Education

The philosophy of the 'visionary', Mark Weiser, was to use the ubiquitous nature of mobile devices to do things for people but, an interesting direction instead would be to utilize these devices to progressively engage them more actively in what

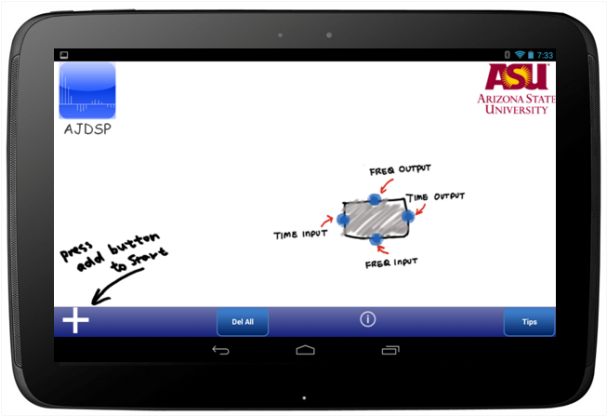


Figure 1.4: Splash screen of the AJDSP app.

they do [37]. Developing apps and methods to use them constructively to augment a course curriculum will assist in negating theories that deem mobile devices to be one of the primary causes for student distractions.

Over the last decade, computer based tools have created a great impact in engineering education. This has led to the development of several tools used in DSP courses that enable students to apply concepts in a variety of contexts. One such important tool is Java-DSP [38, 39], a web-based visual programming environment consisting of various functions built to perform DSP simulations. Recently, mobile devices are being used to progressively engage students through their ability to provide enhanced visual representations of content with interactive user interfaces. Instead of using these devices as an alternative platform for content delivery, exploiting their interactivity can create a personalized learning environment that continues to move outside the classroom [40]. Wireless and mobile technologies can provide learning in multiple contexts [41] by connecting students with their peers and teachers, and introducing instant feedback in the learning process. Studies suggest that students feel more confident, develop critical reasoning and are able to retain their learning for longer when working in such an environment [42]. As against a mere process and spit out of numbers approach, there is a need to train engineers to make them understand

what to sense, how to sense, how to store, how to analyze and process, and how to effectively visualize data.

1.4 Contributions

In this work, we discuss the development of a new scientific paradigm for teaching DSP concepts using an Android app with sensor interfaces. To this end, we have developed Android Java-DSP (AJDSP), an educational graphical programming app to complement instruction in graduate and undergraduate DSP courses. The app can be run on all Android smartphones and tablets, and will be available for free download on the Google Play store by December 2013 [36]. It supports several DSP functions pertaining to topics such as filter design, convolution, multirate signal processing and the FFT.

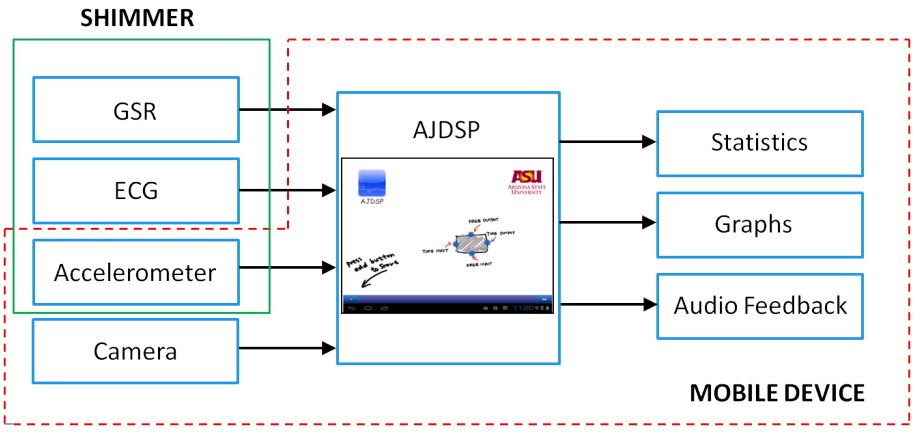


Figure 1.5: Overview of the proposed mobile health monitoring system using Android mobile devices.

The primary focus of this work is to present AJDSP interfaces to wearable wireless sensors, developed by Shimmer Research Inc., and to on-board mobile device sensors (Fig. 1.5). Software modules have been developed and integrated into AJDSP to: (a) acquire data from on-board and SHIMMER sensors; (b) extract relevant features from the sensor measurements; and (c) compute and visualize physiological

parameters. The sensors interfaced with AJDSP include the microphone, camera, electrocardiogram (ECG), galvanic skin response (GSR), and accelerometer. In particular, several algorithms were implemented to analyze the corresponding sensor signals. Parameters such as heart rate and oxygen saturation in the blood have been estimated using the Photoplethysmogram (PPG) signal. Features such as heart rate variability, R-R interval and heart rate vector have been generated using the ECG signal. The number of steps taken by a subject using the accelerometer signal has also been been estimated. In addition, numerous laboratory exercises and tutorials to be deployed in a freshmen engineering course (FSE 100) and an undergraduate level DSP course (EEE 407) at Arizona State University (ASU) have been designed. Furthermore, two interactive workshops were conducted to evaluate the usefulness of the proposed m-Health modules in augmenting DSP education.

In Chapter 2, an overview of the AJDSP app and its various signal processing functions are highlighted. Detailed descriptions regarding the system design, app architecture and signal flow are also presented.

In Chapter 3, details on the SHIMMER sensor hardware used in this work is provided, and a brief background on the physiological signals, namely: ECG, GSR, and PPG is also given.

In Chapter 4, some of the basic Android-DSP modules developed to interface with the sensors to acquire data have been presented. For each module developed, its functionality is described along with snapshots of the user interface (UI).

In Chapter 5, detailed descriptions on the wavelet processing modules are provided. In addition, algorithms implemented to estimate physiological parameters such as heart rate and oxygen saturation have been described. Details on the module to extract ECG features and another module to counts steps have also been presented. The accuracy of the implemented algorithms have been discussed.

In Chapter 6, the laboratory exercises designed to demonstrate health monitoring applications and the role of DSP in them have been presented.

In Chapter 7, details on the methodology adopted to evaluate the usefulness of the AJDSP software in augmenting DSP education is presented. Descriptions about the assessment workshop and its corresponding outcomes are provided.

In Chapter 8, conclusions on the work presented in this document and its scope for the future have been discussed.

Chapter 2

THE AJDSP APP

2.1 App Overview

In this section, we provide a brief background on the AJDSP app and its functionality (Fig. 2.1). AJDSP is a standalone mobile graphical programming app designed to create an appealing learning environment for signal processing students. The app comprises of a block-based interface with each block/module comprising of an input and output signal. The modules are connected together to set-up simulations which enables presentation of DSP concepts in an easily understandable manner. Using AJDSP, students can establish and run DSP algorithms with various configurations on their Android mobile phones and tablets. Laboratory exercises have been developed specifically for the tool that students complete as part of their coursework at Arizona State University. The content of the AJDSP courseware is enriched by incorporating several online multimedia tools such as YouTube videos, interactive

Figure 2.1: The AJDSP educational app for Android mobile phones and tabets.

presentations, social media discussion forums, and a rich suite of documentation. The signal processing modules in AJDSP are broadly divided into signal acquisition and signal processing units. AJDSP has capabilities to generate deterministic and random signals, as well as musical instrument digital interface (MIDI) and dual tone multi-frequency (DTMF) waveforms. Along with a rich suite of time and frequency domain signal processing functions, algorithms such as the fast Fourier transform, filter design, and z-domain operations have been incorporated. Interactive demonstrations to teach concepts of continous-time and discrete-time convolution, and relationships between the z-domain and the frequency-domain using the pole-zero placement method are provided. Furthermore, a new set of sensor and m-Health modules have been developed to acquire data from external and on-board sensors and estimate vital physiological parameters. The interactive nature of the AJDSP app facilitates a stimulus-response cycle in which the problem presented acts as a stimulus and the learners response contributes to the solution provided by the application. This form of feedback reinforces learning and creates a platform for a collaborative and life-long learning. It creates an informal learning environment since problem simulations and exercises can be performed in any real-life setting.

The AJDSP app features can be summarized as follows:

- Graphical programming experience.
- Easy to use multi-touch and drag-and-drop interface.
- Stand-alone tool without the need for internet connectivity
- Easy to install and compatible with all API versions of Android.
- DSP functions include FFT, filter design, multi-rate processing, and wavelets.
- Interface to on-board and external sensors.
- Compact, efficient and scalable.
- Book with J-DSP examples available along with a PC version of the software.

2.2 System Design and Architecture

The system design of the AJDSP app follows the popular Model, View, Controller (MVC) paradigm, where the control, data processing, and user interface components are implemented separately. This method enables easy management of large-sized software projects, where debugging and code optimization can otherwise be con-

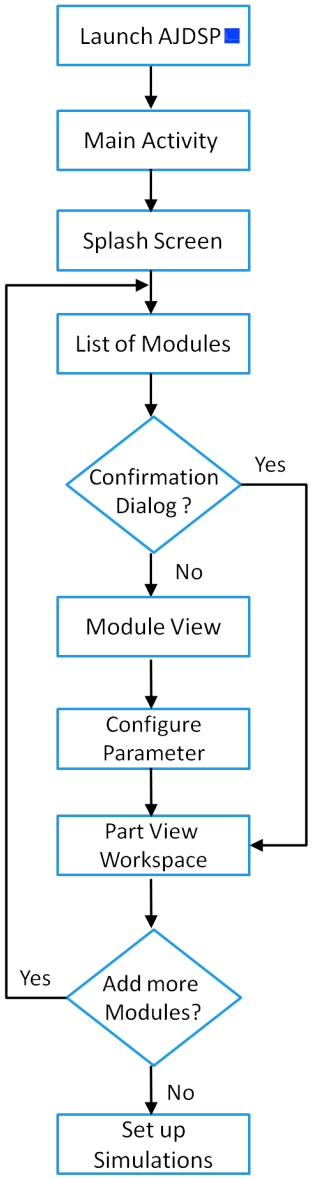


Figure 2.2: Flow diagram illustrating the working of the app as a sequence of steps from launch to DSP simulation.

fusing. Java classes that are implemented to act as the model contain code for data analysis and processing, classes that act as the view involve functions that render graphics on the device, and classes defined to act as a controller contains code that takes care of the control flow of the app. Fig. 2.2 illustrates the control flow of the app as a sequence of steps from its launch to setting up a DSP simulation. When the AJDSP app is launched, the main activity is called, followed by the splash screen which contains buttons that provide basic information on how to use the software. The list of signal processing modules that exist within the app can be accessed from the splash screen. If the selected module does not have a configurable parameter view associated with it, the module is directly added to the workspace, else the parameter view is loaded. The module parameters can then be tuned based on the simulation, and once configured, the module is added to the workspace. This process is repeated to add more modules, and connections are established between them to set up a simulation.

The AJDSP architecture consists of four critical classes, namely: *Part List, Part View, Part Object*, and *Parts Controller*. The relationship between these classes is depicted in Fig. 2.3. The *Part List* class is used to display the list of modules contained in AJDSP. When an item from the list is selected, either the parameter view of the corresponding module is loaded, or a confirmation dialog is displayed. In the former case, the activity associated with the module is started. While, in the latter case, critical information about the module is stored in a bundle data structure and the new module described by the bundle is created and added to the workspace. The module activity is the class that creates a display that users are allowed to interact with and visualize. All the widgets that associate with corresponding module parameters are initialized here. In addition, the module activity encloses the implementation of all the algorithms and data processing. It is also responsible for configuring the

output pin of the module with the necessary signal. Each module consists of a set of vital information in the form of key-value pairs stored in a bundle structure. Some of the information in the bundle include number and type of I/O pins on the module, the signal magnitude, phase, frame number, length, name, and number of the module. The activity runs on the main user interface (UI) thread, hence computationally intensive tasks must be implemented on background threads to avoid stalling the responsiveness of the display. The *Part Object* class contains detailed information on the inherent specifications of a module object such as number of pins, locations of pins, and type of signals (time/frequency domain) that it can process. It encloses all the methods necessary to use this information to create a module object and keep a track of the list of modules created and the connections between them. The module is created using the method *createBlock()*, if the module is already created, the method *updateBlock()* is called. These two methods are defined in the *Parts Controller* class, which is primarily responsible for communicating the data between the key classes and manages the control flow between them. The *Parts Calculator* mainly runs the algorithms that define the functionality of the module, and every module consists of a dedicated custom implementation of the calculator. If a module is connected to other child modules, the *updateChildren()* method defined in *Part Object* is used to systematically get access to the child. Once the control is passed to the child module, the steps to calculating the module output, updating the module information, and checking to see if it has a child, are repeated. When there is no other child module to update, the control is passed to the *Part View* class that displays the workspace where the simulation block diagrams are constructed.

The AJDSP modules were initially constructed to process only 256 samples of a signal. With the addition of the sensor and m-Health modules, the input signals to be processed now had variable lengths. To accommodate for such variability, while

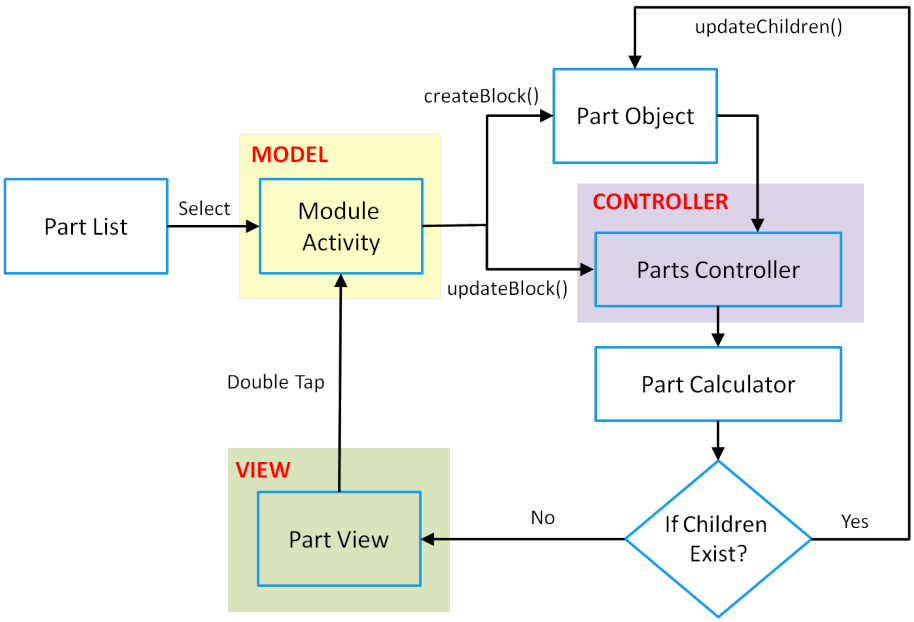


Figure 2.3: Flow diagram illustrating the architecture of AJDSP.

keeping the implementation of each module constant throughout the app, a frame-by-frame processing feature was developed. The paradigm of processing a signal as frames of data has been popularly used in several signal processing applications. The implementation of this feature required keeping a constant update of both the current frame that is being processed and the total number of frames contained in the signal. This information was packed into the inherent key-value data bundle of each module which is communicated to every other module that participates in a simulation.

Chapter 3

SENSOR HARDWARE AND BIOSIGNALS

3.1 Wireless Biosensors for m-Health

Typically, sensor platforms contain a sensing device, an operating system (OS), and communication and power management elements [43]. Interfacing sensors and mobile devices enable the development of a new set of applications that utilizes low-level sensor data and high-level events, contexts and activities inferred from them. Two widely used mode of wireless communication in such a set-up include Bluetooth and Wireless-Fidelity (Wi-Fi). Bluetooth is a short range (upto 10 metres) wireless technology that is low in power, processing and data rates. While, Wi-Fi has a longer range of upto 1000 feet and uses higher power and processing speed to provide a stronger connection, however, both of them operate in the 2.4GHz band. Several low power wireless biosensors are available off the shelf that can be used to measure various physiological and health-related parameters. For example, the HxM BT Heart-rate Monitor is a sensor that can be used to track heart rate, speed, and distance. It supports Bluetooth transmission, is USB rechargeable with a 26-hour battery life, and costs about seventy nine dollars. The Withings WiFi body scale is a sensor that records weight, body mass, and body mass index (BMI) automatically. It costs about two hundred and fifty dollars. The My Glucose Health sensor is a Bluetooth enabled blood glucose meter that has the ability to communicate data online with the clinicians in real-time. This sensors costs about ninety dollars. The Zip FitBit Activity Tracker is a wireless sensor that tracks steps, distances, and calories burned during physical activity. It allows the recorded statistics to be synced to personal

computers (PCs) and mobile devices and costs about sixty dollars. In this work, we use the Sensing Health with Intelligence, Modularity, Mobility and Experimental Reusability (SHIMMER) sensors, a small wireless low-power sensor platform, that can record and transmit physiological and kinematic data in real-time.

3.2 The SHIMMER Sensor Platform

Sensing Health with Intelligence, Modularity, Mobility and Experimental Reusability (SHIMMER) is a small wireless low-power sensor platform that can record and transmit physiological and kinematic data in real-time (Fig. 3.1) [44]. This sensing platform has been adopted to increase the applications of sensor technology in healthcare, using open standards to achieve this goal. The platform focuses on integrating hardware and software components to allow for rapid prototyping of research

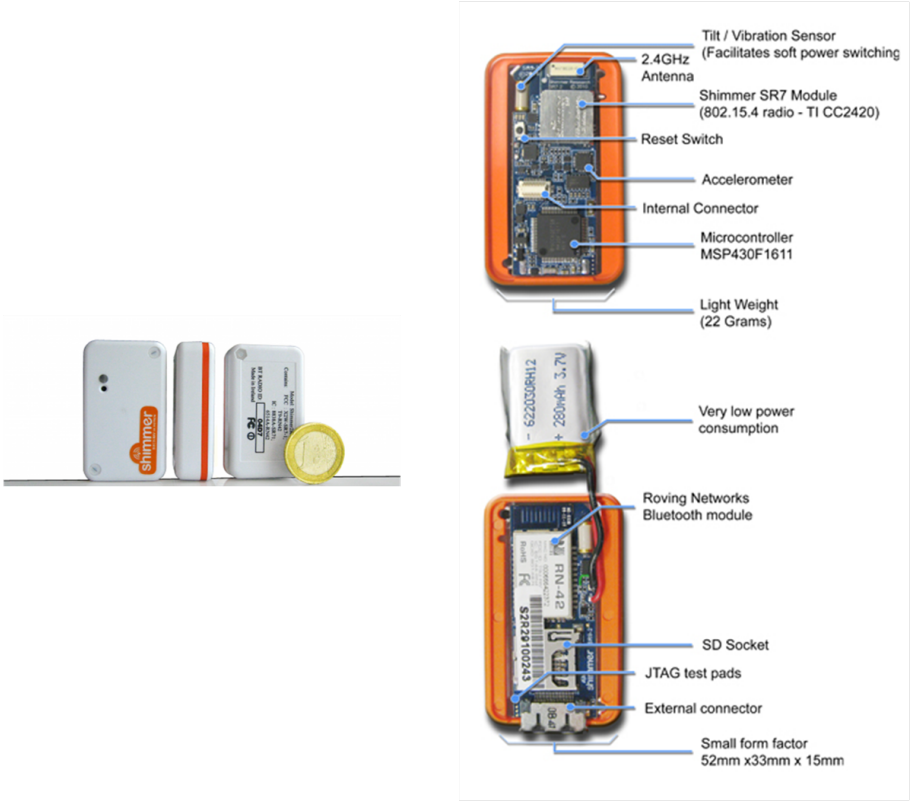


Figure 3.1: The SHIMMER sensor platform.

in biomedical applications [24]. The SHIMMER baseboard forms the core component of this platform and comprises a microcontroller, a Texas Instruments MSP430 MCU, the advantage of which is that it consumes low power when inactive and is effective in medical sensing applications. The baseboard also consists of a Bluetooth or 802.15.4 low power radio transceiver, an accelerometer for activity monitoring, data storage using a micro SD card, and connection capabilities to different kinds of daughterboards (biosensors). The daughterboards extend the abilities of this platform by providing various kinematic, ambient and physiological sensing functionalities: 1) Kinematic sensing: Accelerometers, gyroscopes and magnetometers are included and utilized for inertial measurement applications. 2) Physiological sensing: Incorporates Galvanic Skin Response (GSR), Electrocardiogram (ECG), and Electromyography (EMG) sensors. 3) Ambient sensing: Comprises of temperature and light sensors.

In the recent past, many m-Health applications have used the SHIMMER sensor platform to reliably obtain sensing measurements in real-time for signal monitoring and diagnosis. BioMOBIUS is an example where SHIMMER is used as part of a medical research platform [45]. Accelerometers have been used to study energy expenditure estimation in patients with Rheumatoid Arthritis [46]. ECG sensors have enabled a wavelet-based real-time delineation and compressed sensing-based compression of ECG signals [47, 48]. Motion analysis of patients with Parkinsons disease, stroke, and epilepsy, using a wearable wireless sensor platform called Mercury has been developed in [49]. A combination of physiological sensors has been used to perform activity aware stress detection [50]. In the proposed interface, we employ the SHIMMER platform for real-time sensing, as it includes libraries for app development on Android devices. Moreover, the compact size, extensive sensing capabilities, wearability and light weight nature, make the SHIMMER ideal for creating mobile and nonintrusive physiological signal monitoring systems. The purpose of building this

platform has primarily been for use in various biomedical and activity aware applications, however, it has not been used for demonstrative purposes in an undergraduate DSP course.

3.3 Physiological Signals - a background

3.3.1 Electrocardiogram (ECG)

An ECG device detects and amplifies the small changes in electrical activity of skin when the heart muscle depolarizes at each heart beat. When heart is at rest, the muscle cell has a negative membrane potential which is depolarized and decreased to zero with an influx of cations, namely: Sodium (Na) and Calcium (Ca). This creates a wave of depolarization and the electrical impulse is generated in the sinoatrial (SA) node of the right atrium and spreads a wave through the atria before arriving at the atrioventricular (AV) node. This wave further spreads to all parts of the ventricles through the His-Purkinje system. The signal is thus detected as a rise and fall in voltage between electrodes placed on either side of heart. These electrodes can be combined into pairs such as:

- LA + RA (Left arm, Right arm)
- LA + LL (Left arm, Left leg)
- RA + LL (Right arm, Left leg)

The output signal from each of these pairs is called a lead, and each lead provides a different perspective of the heart with respective to the angle of sight. The lead configurations can be summarized as the following:

- Lead I: voltage between (positive) RA and (negative) LA electrodes = RA - LA
- Lead II: voltage between (positive) LL and RA electrodes = LL RA
- Lead III: voltage between (positive) LL and LA electrodes = LL LA

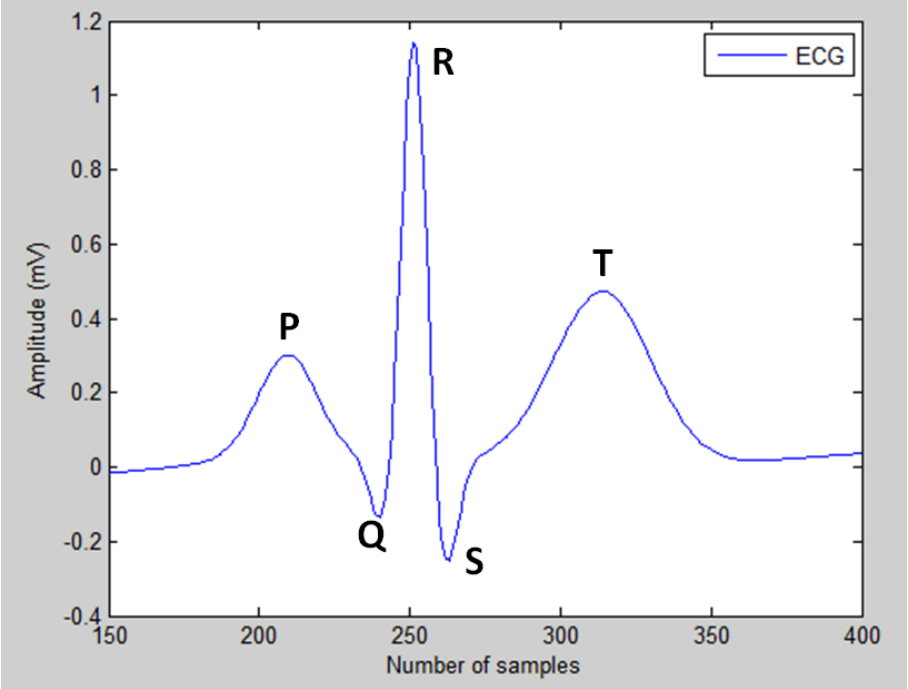


Figure 3.2: An ECG sample obtained from Physionet marked with its P, QRS-complex, and T points.

The ECG signal is used to observe the electrical activity of the heart over a period of time. The measurement provides valuable insights into the functionality of the heart based on the regularity of the heartbeats. A sequence of normal heart beats is knows as normal sinus rhythm. All normal beats originate in the sinoatrial node, while abnormal beats (arrthythmia) can originate in the atria, ventricles, or atrioventricular node. An ECG waveform for a single heart beat typically comprises of three significant wave segments, namely: the P-wave, the QRS-complex, and the T-wave (Fig. 3.2). The P-wave corresponds to wave of depolarization that starts in the atria of the heart, whose amplitude is usually 1/5th of the R-peak. The QRS-complex, a tall and narrow segment of the wave, corresponds to electrical excitation of the two ventricles and its depolarization. The T-wave corresponds to repolarization of ventricles, which is the recovery stage of the ventricles thereby restoring it to rest state, and has an amplitude about 1/3rd of the R-peak. Typically, the atrial (P-P) and venticular (R-R) rate must

be the same, but sometimes there may be discrepancies due to the conditions of the myocardium. In general, an ECG recording of atleast a ten second period must be analyzed to assess its regularity.

$$HeartRate = \frac{60}{R - Rinterval} \tag{3.1}$$

The heart rate is estimated using (3.1), where the time interval between successive R-peaks is known as the R-R interval. Normal resting heart rate is said to be in the range of 60-100 beats per minute.

3.3.2 Galvanic Skin Response (GSR)

The GSR sensor measures the electrical resistance of skin based on the amount of moisture content present in it. This conductance varies based on skin and muscle tissue responses to stimuli. It is used to detect stress, fear or anxiety, all of which make sweat glands more active and cause a decrease in the skin resistance. Two signal characteristics that can be visualized from the GSR signal are: (a) Skin Conductance Level (SCL) a slowly varying signal; and (b) Skin Conductance Response (SCR) a fast varying signal. Important features extracted from this signal are the amplitude and latency of SCR and the mean and standard deviation of SCL. The distribution of the SCL peak and the SCR peak rate can be shown to carry information regarding the stress level of a person [50]. Using these features, classification of skin conductance into Tonic conductance and Phasic conductance can be demonstrated. The former is observed as a sharp transition in the signal values due to a stimulus and the latter is observed 1-2 seconds after the stimulus. Tonic conductance can further be shown to increase with stress levels or demanding mental activities. In addition to this, concepts of a startle response, which is the physiological response of the body due to a sudden stimulus, can be presented. The total number of startle responses in a

windowed segment of GSR, the sum of the response magnitude and the sum of the response duration can be shown to characterize the startle response.

3.3.3 Photoplethysmogram (PPG)

The PPG is a physiological signal that provides a volumetric measurement of organs, such as the heart and the lungs. The word 'photo' in photoplethymogram indicates light, while plethysmos, a Greek originated word, corresponds to an increase. The PPG signal is generated based on the differential absorption of light by the human skin. A pulse oximeter is a sensor that measures the PPG to estimate parameters such as heart rate and oxygen saturation in the blood. It consists of two LEDs, one generating red light (660nm) and the other infrared light (940nm), and one detector (photo diode). The detector can be placed either at the side of the LEDs, thereby providing a reflective plethysmography, or across the LEDs for transmission plethysmography. The PPG signal typically comprises of two components, a pulsatile component and a constant component obtained from the measurement of changes in absorption of the incident light by the skin. The pulsatile components are a result of the light absorption by the blood which indicate variations corresponding to the volume of blood in the blood vessels present below the skin. While, the constant component is a result of the light absorption by other surrounding tissues and the bone. The PPG can be used to monitor heart-rate and cardiac cycle, respiratory rate, blood pressure, oxygen saturation and hypo- and hyper-volemia.

Chapter 4

BASIC SENSING MODULES

4.1 Introduction

To better understand the role of DSP in real-world applications and gauge the corresponding challenges associated with them, data used in the learning process must resemble the ground truth as closely as possible. When students are provided with data whose source and structure are somewhat perceivable, it motivates them to think in terms of the application context to obtain solutions to problems. As part of the efforts made towards demonstrating such DSP applications using AJDSP include the development of a strew of data acquisition modules. In addition, significant changes to the architecture of the app primarily concerning the data flow process have been made. In many signal processing domains, it is a common practice to decompose the original data into windows or frames of a signal for further analysis and processing. The architecture of the AJDSP app has been modified to accommodate the feature of frame-by-frame processing of the input signal. As part of the framing scheme, a systematic method has been created to seamlessly pass data packets that contain critical information required for the appropriate functioning of all the blocks. For the purpose of data collection, several signal acquisition modules have been developed to provide users access to the different sensors present on board and in the Shimmer platform. The native Android API is used to connect to the microphone, camera and accelerometer; while a Bluetooth connection is established to communicate to the external Shimmer sensors. Furthermore, for ease-of-use while performing laboratory exercises, modules with built-in sensor data such as audio/noise signals and ECG

signals have been developed. The user-interface designed for this category of modules provide controls to start and stop data streaming, visualize data continuously as frames, and has menu options to choose the block output. In addition, there is a provision to change custom signal parameters associated with each module. Once the required data has been recorded, the modules can be added to the app workspace for data processing. In the case of data acquisition from Shimmer sensors, users must disconnect the Bluetooth connection before navigating to the workspace to preserve battery life of the device during app usage.

4.2 Long Signal Generator

The Long Signal Generator block comprises of audio and noise signals each of 8192 samples in length. The data can be visualized as frames of 256 samples each. Buttons are provide to navigate across the signal frames. Parameters such as gain, frame size, and amount of overlap can be specified. The gain can take any value between 0 and 1000; frame size can be varied between 1 and 256; and the amount of overlap expressed as a percentage can take either of the values 0, 50, or 100 (Fig. 4.1). The architecture of AJDSP allows processing of data only 256 samples at a time. Hence, this block comes with a feature of parsing the signal one frame at a time when connected to other blocks in the workspace.

4.3 Sound Recorder

Speech processing is one of the most widely researched area by the signal processing community that continues to be embedded with several unsolved problems especially with respect to speech recognition. Functions in AJDSP illustrate speech signals in terms of their spectra, and features such as energy density, frequency, and voiced/unvoiced segments by recording audio data in real-time. The Sound Recorder

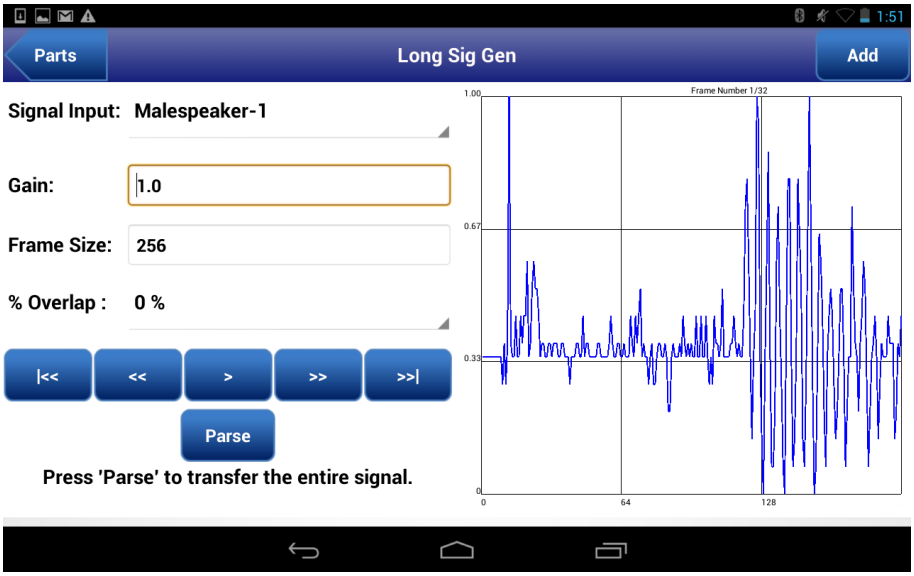


Figure 4.1: Long Signal Generator block displaying voiced segments of a male speaker.

is one such block that allows access to the microphone on the mobile device to record upto ten seconds of data at a sampling rate of 8000 Hz. The size of the data frame can be adjusted to be between 1 through 256 samples. The block also contains buttons in the view to navigate to a specific frame of data (Fig. 4.2).

4.4 Accelerometer

The accelerometer can be used to measure instantaneous acceleration due to forces acting on the sensor. Analysis of accelerometer data requires extraction of features such as mean, standard deviation, and energy for each axis, and correlations between axes. Students can be provided with an insight on how physiological responses tend to be affected by physical activity. Examples include, displaying readings and waveforms of sensor measurements obtained while a person is at rest, in comparison to while a person is walking. Therefore, a need to incorporate accelerometer measurements in context-aware applications such as stress detection can be demonstrated. Furthermore, the appropriate placement and positioning of sensors

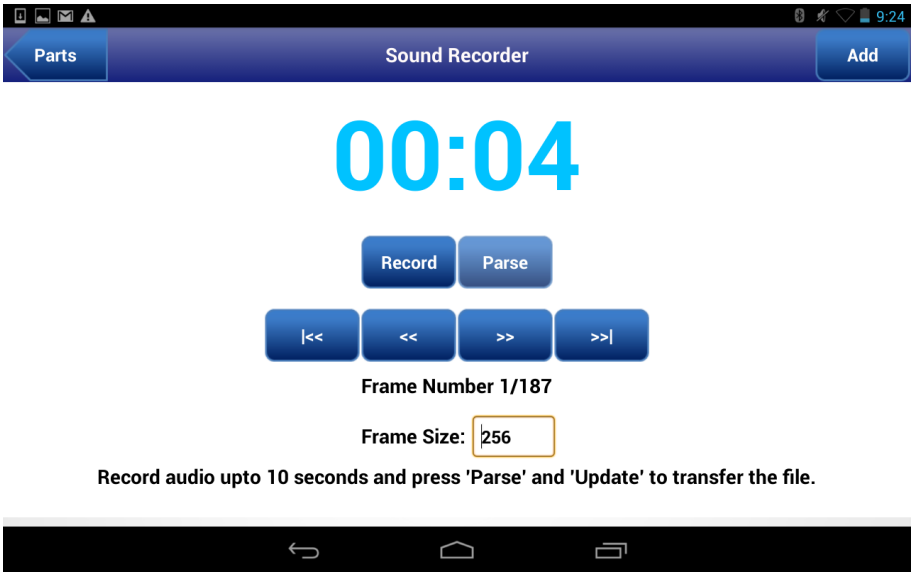


Figure 4.2: User interface of the Sound Recorder block.

for different appplications can be presented. For example, it can be shown that placing the accelerometer on the hip helps in classifying physical activity [51]. AJDSP has functions to interface with both the on-board accelerometer and the external Shimmer accelerometer which are discussed as follows.

4.4.1 On-board Accelerometer

The native Android API can be used to continuously stream X, Y and Z-axis data from the accelerometer whose instantaneous values are displayed both as numbers and a plot in the user interface of the block as shown in Fig. 4.3. This block also gives an example application of accelerometers by demonstrating their role in a step counter. The accelerometer magnitude is calculated from three axis signals and, is used to count the number of steps taken. In addition, estimates of the duration of one of three activities namely, walking, running, and standing are also displayed.

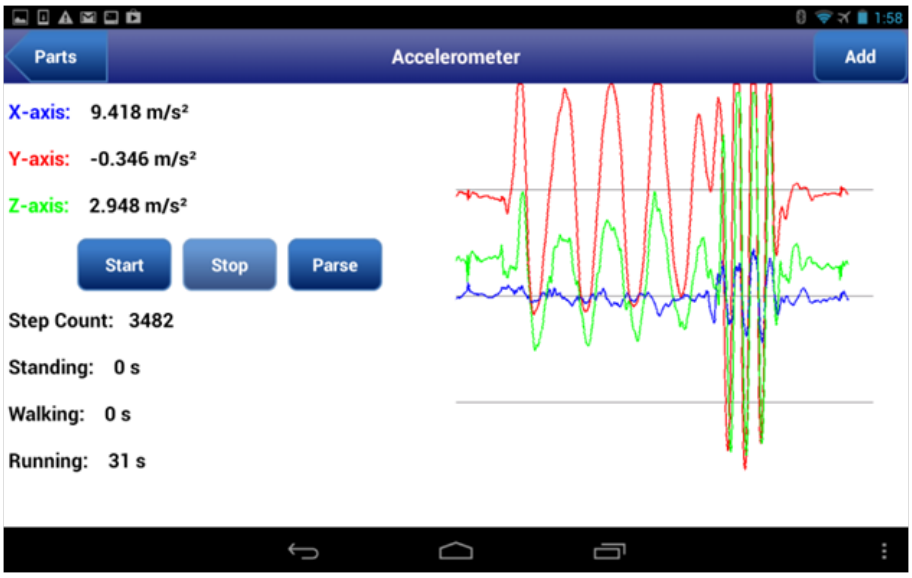


Figure 4.3: User interface depicting the streaming of data from the built-in accelerometer.

4.4.2 Shimmer Accelerometer

Data from the accelerometer contained in the Shimmer baseboard can be acquired via Bluetooth using the Android library developed by Shimmer Inc. The user interface consists of buttons that enable establishment of a connection between the mobile device and the sensor easily. The 'Connect' button opens a dialog with the list of devices that are in the vicinity of the mobile device with a Bluetooth pairing capability. Toast messages to indicate the success/failure of the connection are displayed on selecting the Shimmer sensor from which data packets are to be acquired. The Shimmer baseboard unit may contain either one of the other sensor daughter boards (ECG/GSR) mounted in it apart from the accelerometer. The 'Configure' button is used to configure the baseboard to transmit data from the appropriate sensor as required by the simulation at hand. Within this block, by default the accelerometer is configured for data transmission on tapping the 'Configure' button. The 'Start' and 'Stop' buttons can be used to start or stop the process of data streaming and

Figure 4.4: Shimmer accelerometer view depicting the transitions in the X, Y and Z-axis signals based on the position of the sensor with respect to the direction of the earth's gravitational force.

the corresponding signals can be visualized in real-time (Fig. 4.4). In addition to buttons to navigate through the data frames, the block output can be selected to be either the magnitude, X, Y or Z-axis signal (Fig. 4.5).

4.5 Biosignal Generator

A part of the goal in this work was to provide students pursuing DSP courses to work with as close to real data as possible. The Shimmer biosensor interfaces in AJDSP are especially aimed at giving a hands-on experiencing with acquiring physiological signals. Data can be streamed in real-time from the SHIMMER-based electrocardiogram (ECG) sensor and galvanic skin response (GSR) sensor. Since, obtaining measurements from every subject prior to performing a laboratory exercise may be cumbersome, a block containing open source ECG data for normal and abnormal health conditions has been developed. Using these blocks physiological signal characteristics are visualized and related to medical conditions.

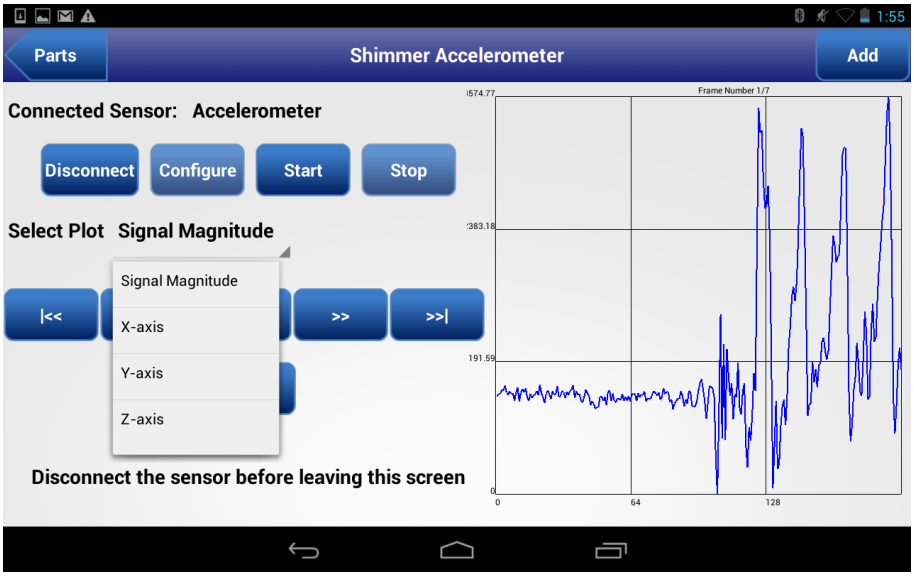


Figure 4.5: Shimmer accelerometer data frames with the pull down menu to select block output.

4.5.1 Pre-loaded Physiological Signals

This block closely resembles the Long Signal Generator containing ecg data instead of audio data. In addition, the block estimates the heart rate of the signal by detecting the R-peaks using Daubechies 06 Wavelets (Fig. 4.6). Details on the algorithm implemented for this estimation is provided in Chapter 5. The ecg data can be obtained from the sources listed in [52]. The block comprises of eight signals, of which five are normal and the remaining three signals depict a certain abnormality. Among the normal signals, there is one synthetic signal [53], and noisy versions of two other signals. Medical conditions such as Atrial Fibrillation, Ventricular Tachycardia, and Ventricular Fibrillation are the abnormalities present in the data. The diversity in the nature of the data will provide students with a broad perspective on ecg signals and their characteristics.

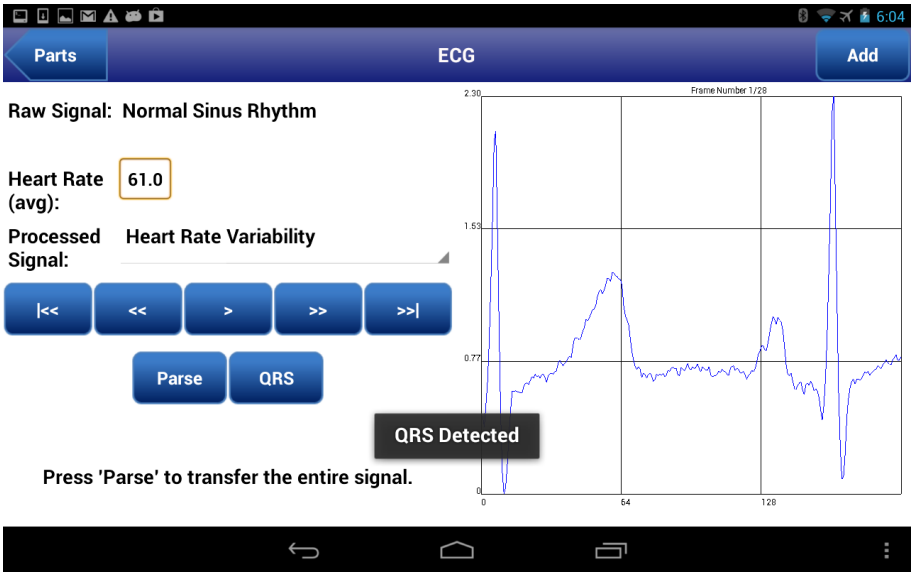


Figure 4.6: ECG Signal Generator block displaying a normal sinus rhythm.

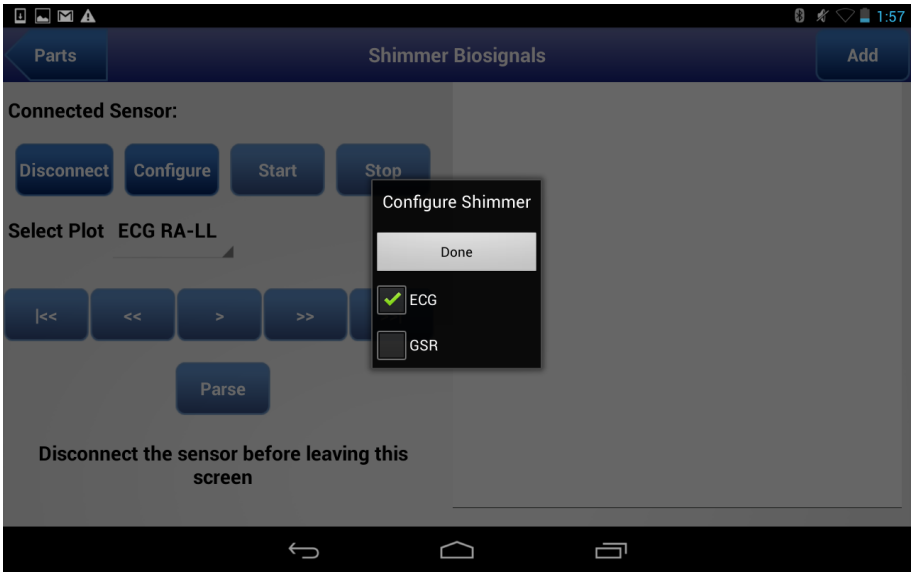


Figure 4.7: User interface to configure the Shimmer ECG sensor.

4.5.2 Shimmer ECG/GSR Sensors

This block allows wireless connection to shimmer ECG/GSR sensors to stream the corresponding packets of data via Bluetooth. Once the sensor is placed on the chest/wrist using straps, they are configured using the 'Configure' button and checking the required option as shown in Fig. 4.7. The Shimmer platform has the capability

of recording ECG signals in either Lead I, II or III configurations. The electrodes are used to make contact between the subject and the sensor. Data is streamed into the app and an there exists an option to observe frames of either lead I (LA-LL), lead II (RA-LL) or the skin response signal. The sensor has to be disconnected before navigating to the workspace to process the acquired data.

Chapter 5

WAVELET PROCESSING AND ALGORITHM MODULES

5.1 Introduction

Signal processing research and education using mobile devices have seen a large surge of interest in the recent past. Several research groups in the world are looking to use mobile devices as diagnostic tools to detect and track the state of well being of its users. A few of the signals that are keenly studied include the electrocardiogram (ECG), electromyogram (EMG), photoplethysmogram (PPG), respiratory signals, electronencephalogram (EEG), and galvanic skin response (GSR).In this chapter, the modules developed to demonstrate non-invasive health monitoring applications using Android mobile devices are presented. Algorithms implemented to make estimations from the sensor data, namely ECG, accelerometer signal, and PPG are described. The modules can be used as stand-alone demonstrations or be connected to other DSP modules in AJDSP. Discussions on the performance of the implemented algorithms where applicable, are enclosed. Scope for improvement of the accuracy of the estimations are also provided.

5.2 Discrete Wavelet Transform

5.2.1 Background

Signals whose frequency content do not change with time, meaning they have all frequency components in it existing at all times, can be easily analyzed using the popular Fourier transform (FT) technique. However, most real-world signals such as speech, ECG, and EMG do not fall under this category, and have varying frequency

components with time. The limitation in the FT in providing the exact time instants at which the frequency components exist in a signal made way for a more powerful transform to be developed, called the wavelet transform (WT). The WT provides a time-scale representation of data which implies that there is a provision for localization in both time and frequency, where frequency is known to be the inverse of scale in this context. Before dismissing the FT completely for its lack of time resolution of frequency components, researchers proposed the Short-Time Fourier Transform (STFT) which tried to overcome the shortcomings of the FT by analyzing the signal in smaller time windows or segments of the original signal. Of course the assumption is that the signal within that window of time will have a constant frequency component, which would in turn allow us to transform each segment using the FT. The problem that arises in this case is based on Heisenberg's Uncertainty Principle, according to which it is impossible to know what exact frequency component exists at which exact time instant. Instead, it is only possible to get an idea on what band of frequencies are present in specific time intervals [54]. Moreover, in using the STFT, one of the hardest decisions to make is the choice of windows and their width. Narrow windows are said to be more compactly supported and will be able to provide a better resolution in time, but inadequate resolution in frequency. However, a window of infinite support will be able to provide a better resolution in frequency, but a poorer time resolution and starts to resemble the FT. The choice of window will hence depend on the application at hand and the signal we are dealing with.

5.2.2 Multiresolution Analysis

To solve the issue of resolution, a new breed of transform called the wavelet transform (WT) was developed. It allowed for better frequency resolution of low frequency components and better time resolution of high frequency components. Since,

most real world signals have higher frequency components occurring for shorter durations and lower frequency components occurring for longer durations, such an approach proved very valuable in the analysis of these signals. Wavelets are short waves of oscillations that are used to build a basis to execute a transform using scaled and translated versions of itself. They are used to perform a moving correlation with the input signal, similar to the use of sinusoids in the FT. The wavelet function in continuous-time is defined as given by (5.1), where s and τ represent the scaling parameter and translation parameter respectively. The continuous wavelet transform (CWT) of a signal is summarized in (5.2). Two popularly used wavelets in the CWT are the Meyer wavelet and the Mexican Hat wavelet.

$$\psi_{\tau,s}(x) = \frac{1}{\sqrt{s}}\psi\left(\frac{x-\tau}{s}\right) \tag{5.1}$$

$$W_{\psi}(s,\tau) = \int_{-\infty}^{\infty} f(x)\psi_{\tau,s}(x)\,dx \tag{5.2}$$

To reduce redundancy and improve the computation of the WT, the discrete wavelet transform (DWT) was introduced. In this technique, the discrete-time signal is analyzed using a series of high-pass and low-pass filters to obtain time-scale representations of the signal. The technique also goes by the name sub-band coding or multiresolution analysis. Equations (5.3) and (5.4) give the mathematical definition of the discrete wavelet and the DWT. The scale samples follow a geometric sequence of ratio 2, and this property makes it a dyadic wavelet transform [55]. Such a power-of-two logarithmic scaling of both the translation and dilation steps is the most simple and efficient method of discretization, thereby enabling construction of an orthonormal wavelet basis [56]. The building blocks of the DWT are shown in the Fig. 5.1, where the subscript 'a' stands for analysis and the subscript 's' stands for synthesis. Scale in the context of the DWT refers to the number of levels of decomposition of the signal, or the number of times either of the analysis or synthesis blocks are used to transform

the signal [57]. The Daubechies family of wavelets is one of the most popular in the discrete domain.

$$\psi_j[n] = \frac{1}{\sqrt{2^j}}\psi\left(\frac{n}{2^j}\right) \tag{5.3}$$

$$Wf[n, 2^j] = \sum_{m=0}^{N-1} f[m]\psi^*[m-n] \tag{5.4}$$

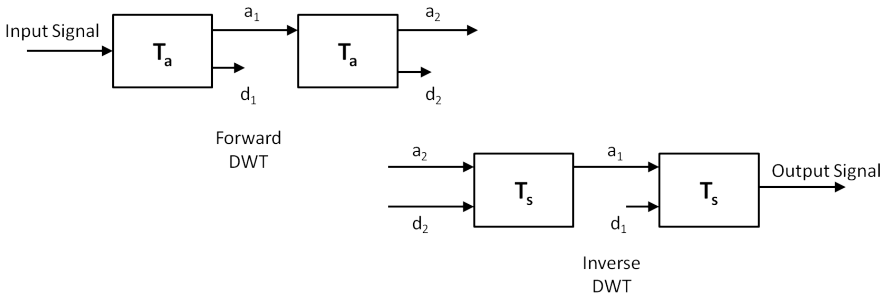


Figure 5.1: Building blocks of the forward and inverse DWT over two scales.

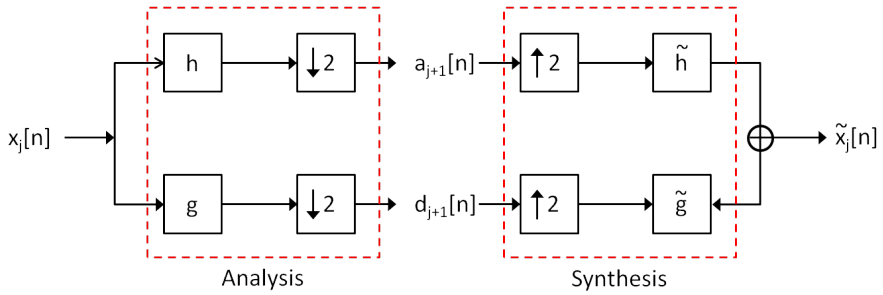


Figure 5.2: Input signal is decomposed using a low-pass (h) and high-pass (g) filter and subsampled. Reconstruction involved upsampling and filtering with dual filters $\sim h$ and $\sim g$.

The signal decomposition is realized using conjugate mirror filter banks that comprise of a filtering operation followed by a subsampling operation as shown in Fig. 5.2. Filtering corresponds to convolving the signal with the impulse response of the filter. Equations (5.5) and (5.6) show the mathematical operation of the filter bank. The filtering alters the resolution of the signal, which is the amount of information, while the subsampling alters the scale. The DWT essentially decomposes

the signal into coarse approximation coefficients (low-frequency components) and detail coefficients (high-frequency components) using the scaling and wavelet functions respectively. Since the decomposition involves successive subsampling, the number of levels a signal can be decomposed into depends on its length, which must be a power of two to enable the dyadic transformation. The output signal from the analysis stage will contain the scaling coefficients concatenated with the detail coefficients from higher scales followed by the detail coefficients of the lower scales. For example, if a signal of length 256 samples (2^8) is decomposed into four levels, i.e. $j = 4$ in equation (5.3), then the output signal coefficients are stacked as shown in Fig. 5.3. The signal can be reconstructed by a process of upsampling followed by filtering

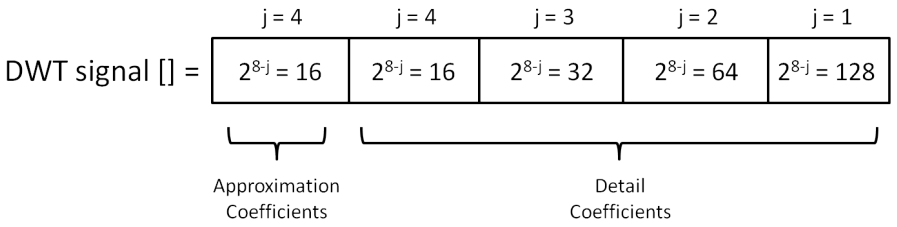


Figure 5.3: A discrete wavelet transformed (DWT) signal array stack of length 256 after a 4-level decomposition.

which comprises of the synthesis stage. The filters employed in both the analysis and synthesis of the signal are similar, with one being a time reversed version of the other. In addition, it is said that if the filters are not ideal half band filters, then perfect reconstruction cannot be achieved.

$$y_{\mathrm{LF}}[n] = \sum_{n} x[n]h[2k - n] \tag{5.5}$$

$$y_{\mathrm{HF}}[n] = \sum_{n} x[n]g[2k - n] \tag{5.6}$$

where LF and HF denote low-frequency and high-frequency respectively.

The forward DWT block in AJDSP (Fig. 5.4) takes an input signal and uses a dyadic transformation to produce a set of output scaling (low-pass) coefficients

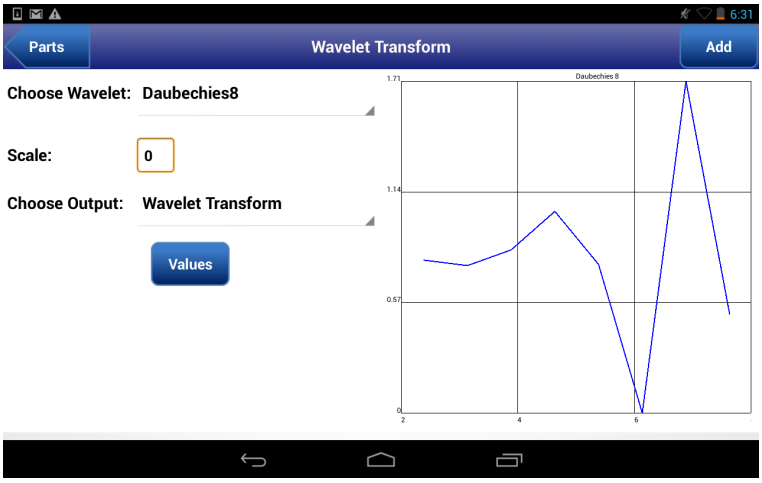


Figure 5.4: Forward DWT module user interface.

and detail (high-pass) coefficients. The open source library jwave was used to implement the transform in AJDSP [58]. The wavelet appropriate for the application at hand, and the number of multiresolution levels/scales for decomposition can be configured. One of the following wavelets can be selected, namely: Haar, Daubechies4, Daubechies6 and Daubechies8, Legendre2, Legendre4 and Legendre6, and Coiflet6. The wavelet coefficients and its plot can also visualized (Fig. 5.5).

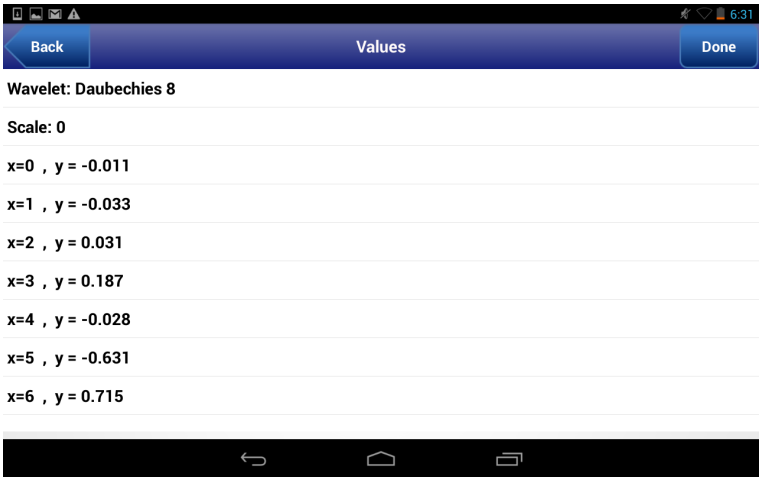


Figure 5.5: Coefficients of the Daubechies8 wavelet.

5.3 ECG Feature Extraction

An ECG sensor detects small changes in the electrical activity of the skin at every heart beat. The measurement is used to analyze the functionality of the heart based on the regularity of heartbeats. An important processing step in ECG signal analysis is the extraction of the QRS-complex and detection of the R-wave peaks [59] (Fig. 5.6). Other features that can be extracted include the mean and standard deviation of heart rate and R-R interval, root mean square (RMS) value of the differences between successive R-R intervals, and percentage of heart beat intervals with a successive R-R difference in interval greater than 50ms (pNN50) [60]. In general, crucial information in ECG falls in the frequency between 0.05 - 50 Hz. Considering even minor changes to the time and amplitude information of the ECG signal has been associated with rigid clinical interpretations, it is important to preserve the magnitude and phase of the ECG during filtering and transform operations. The ECG feature extraction module extracts features such as R-R interval, Heart Rate Variability (HRV), and instantaneous heart rate vector. Standard statistics mentioned above can also be estimated using the *Statistics* module in AJDSP to generate additional features (Fig. 5.7). The discrete wavelet transform approach is selected to delineate ECG signals using the popular Daubechies wavelet family. The tasks performed by the module include peak detection, heart rate estimation, and feature generation.

5.3.1 R-peak Detection

Several methods have been used to identify the R-peaks in an ECG signal. A comprehensive report on the various algorithms in the literature have been reported in [59]. When using wavelets to process a signal, the choice of wavelet function depends primarily on the application. Since the Daubechies wavelet family resembles an ECG

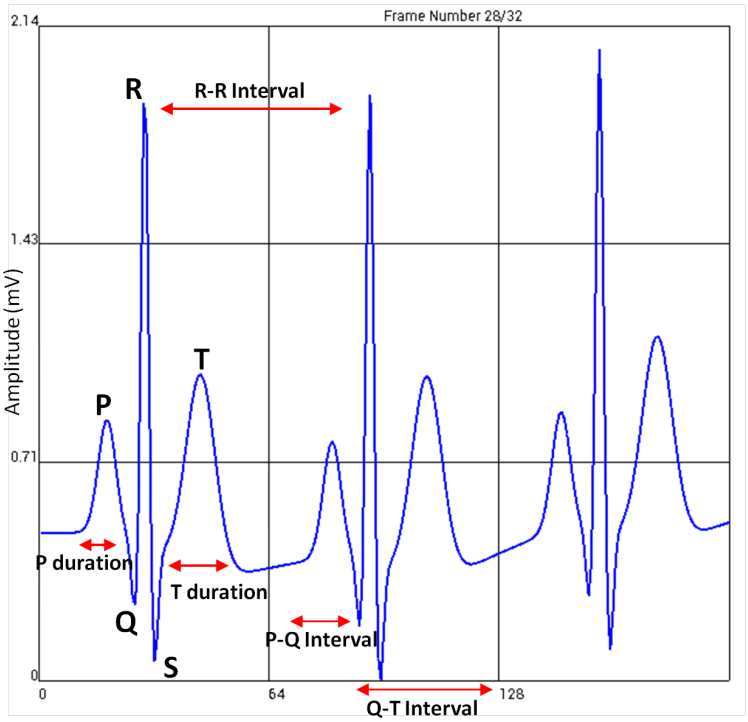


Figure 5.6: A synthetic ECG signal marked with its characteristic wave segments and time intervals.

QRS complex, it is widely used in the analysis of these signals. Although it has a greater computational overhead compared to Haar wavelets, it has the ability to pick up the signal details that are missed by the Haar wavelet. From [61] it is observed that the Daubechies06 (Db06) scaling function better represents QRS complexes than the Daubechies04. The multiresolution analysis of a sample ECG signal using Db06 is depicted in Fig. 5.8. The signal was sampled at 250 Hz, and it can be seen that most of the ECG signal details are contained in the detail levels between 2^3 to 2^5 scale. While, the high frequency noise is picked up in scales 2^1 and 2^2. To detect the R-peaks, the signal is first decompose into 8 levels using the DWT with Db06. Only details between levels 2^3 to 2^5 are preserved and all other details are made zero. This ensures removal of the low and high frequency noise components. The signal is then reconstructed with the reduced coefficients using the IDWT. The resulting signal samples are squared to make the signal transitions more prominent. A peak is

detected at points whose values are greater than 15% of the mean amplitude of the absolute value of the above QRS signal [62].

Using these functions, demonstrations on the different configurations of electrode placement used to acquire the ECG measurements can be provided. Typical characteristics of ECG waveforms like the QRS-complex, P-wave and T-wave segments and their corresponding range of time intervals for normal and abnormal recordings can be understood. ECG signal artifacts such as low frequency baseline wandering and high frequency power line interferences that occur during signal acquisition can be observed. Concepts of signal denoising operations and the challenges faced can be highlighted. In addition, the time-domain and the frequency-domain vi-

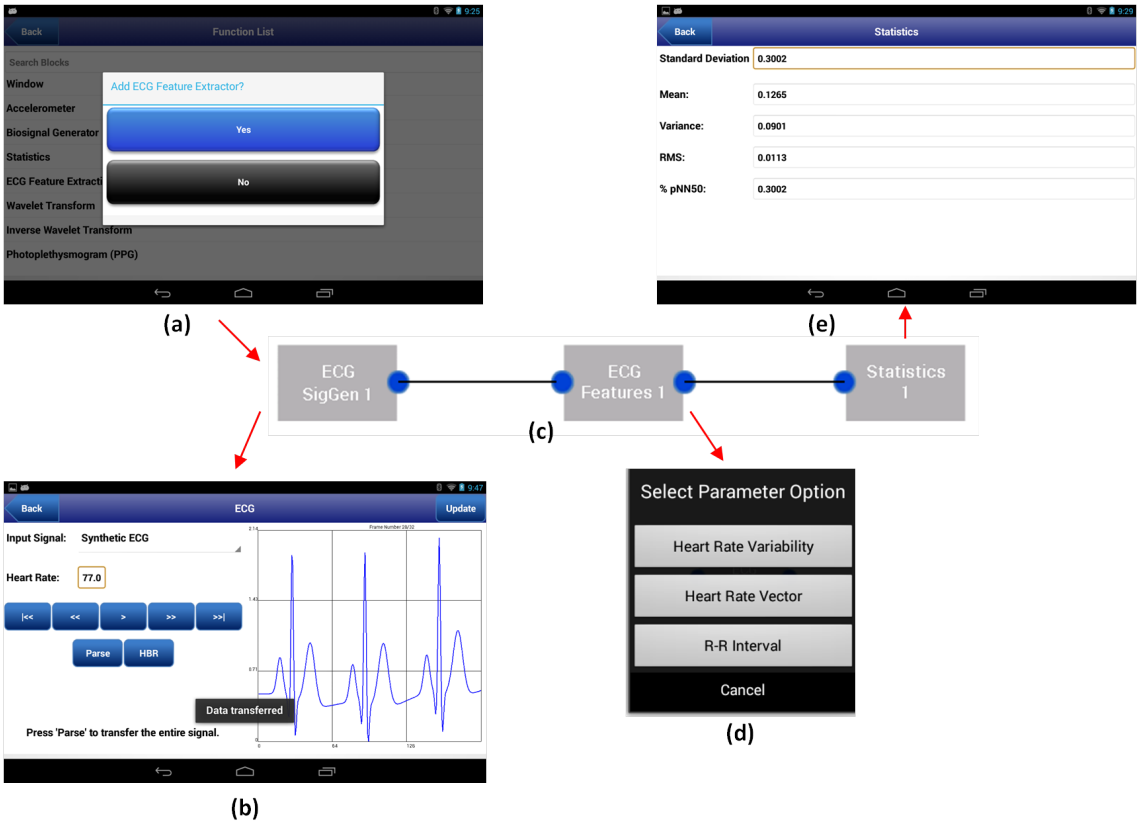


Figure 5.7: A simulation in AJDSP to extract ECG features using DWT. (a) Dialog to select the module from the part list, (b) The ECG signal generator, (c) Simulation set-up in the work space, (d) Parameter option menu, (e) Statistics module view.

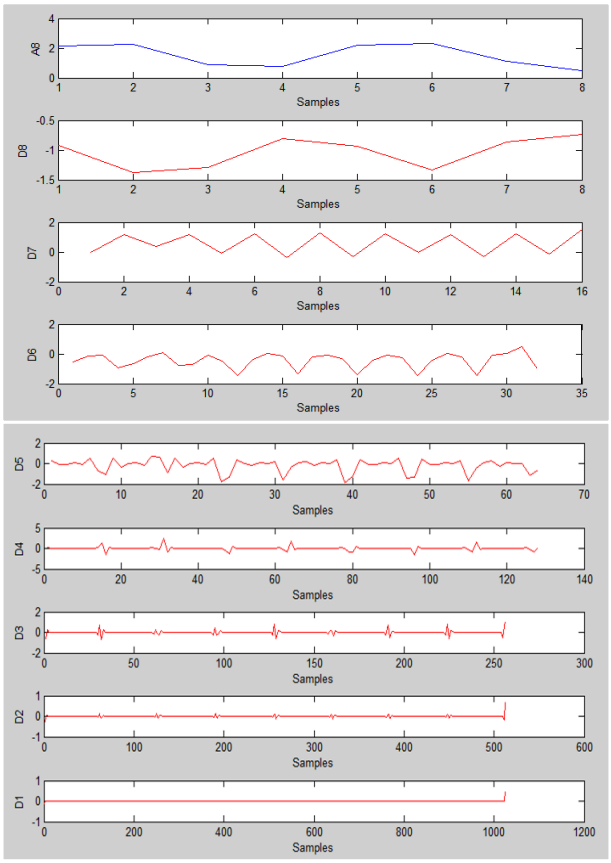


Figure 5.8: The synthetic ECG signal decomposed into 8 scales using Db06 wavelets with each of its detail coefficient signal and the final approximation coefficient signal depicted.

sualization of extracted feature vectors such as, R-R interval and heart rate variability (HRV) can be associated with corresponding health conditions.

5.4 Bio-parameter Estimation from PPG

In this work, the built-in camera on Android mobile devices is used to extract a PPG and estimate physiological parameters such as heart rate and oxygen saturation are estimated. By recording a video of a users finger tip, a PPG signal can be extracted since there is a change in blood volume inside the finger tip in accordance to the systolic and diastolic movement of the heart. This in turn results in variations in light absorption which is reflected by the changes in the average pixel intensities

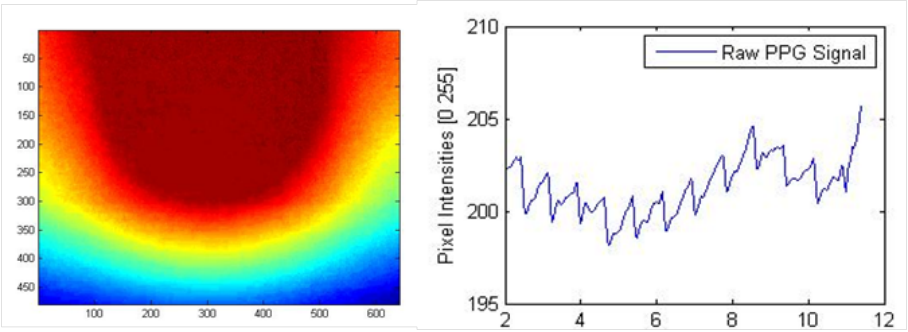


Figure 5.9: The pixel intensities present in a single video frame (left) and the extracted PPG signal (right).

of every frame of the recorded video as seen in Fig. 5.9. Studies suggest that the sensitivity to changes in the sympathetic system is greater in finger tips regions as opposed to regions such as the ear lobes [63].

5.4.1 Heart Rate

Recent efforts to accurately estimate heart rate using the PPG signal involved a variety of techniques. Pelegris et al. employed a local synchronization and matching technique with a crude template signal to estimate heart rate using a mobile phone video recording of a finger tip [64]. While, Poh et al. used Independent Component Analysis (ICA) and source separation to extract the PPG, and employed frequency analysis to estimate heart rate using a video recorded with a computer's built-in webcam [13]. Although these methods have shown reasonable success, there exists a serious challenge in extracting a strong and clean PPG from a mobile device video. This is mainly due to motion artifacts in the video that arise from movements of the finger tip in contact with the camera, thereby resulting in noisy PPG. In this work, an algorithm to estimate heart rate has been developed based on multiresolution analysis. Efforts have been made to get a better estimate of the PPG and to isolate the band of frequencies based on the sampling rate of the Android mobile devices. The proposed algorithm to compute heart rate primarily comprises of three parts,

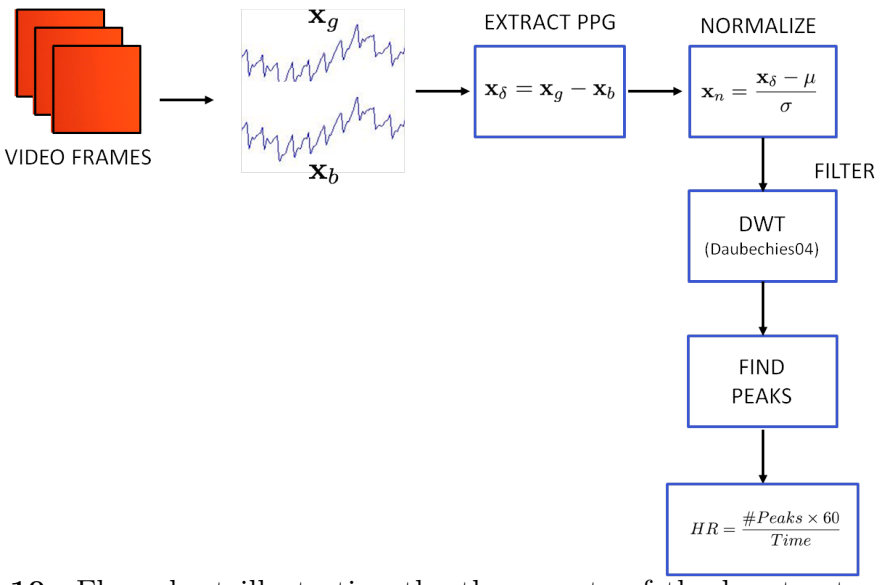
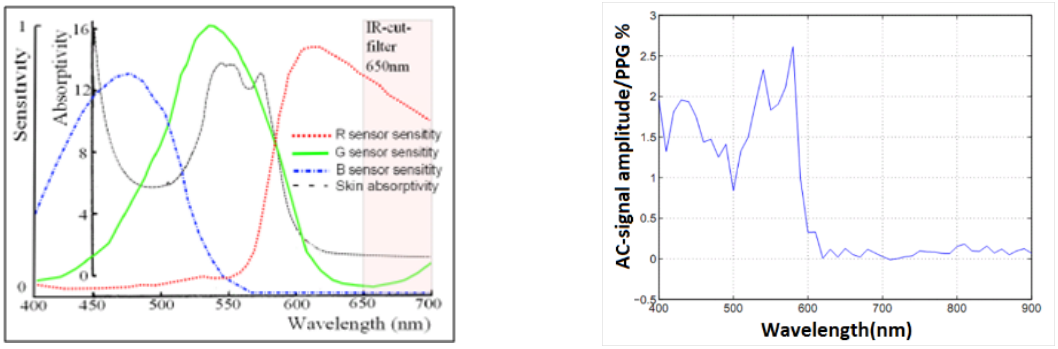


Figure 5.10: Flow chart illustrating the three parts of the heart rate estimation algorithm using the PPG signal.

namely: extraction of the PPG signal, filtering, and peak detection. An overview of the proposed algorithm is illustrated in Fig. 5.10. To extract the PPG signal, the average brightness of every frame for both green and blue channels are computed. x_g and x_b correspond to the raw PPG signal obtained from the green and blue video channels respectively. The choice of these color channels has been motivated by the fact that the human skin shows maximum absorption of light in the green wavelength region, which is depicted by the black line in Fig. 5.11(a). It can be seen that this region coincides with the green sensor's sensitivity of the camera. The region that spans the blue and green wavelengths also have maximum variation in the AC component of PPG as seen in Fig. 5.11(b) [65]. Furthermore, on observing the spectrum of light absorption by the skin for green, red and blue wavelengths, the green spectrum shows the highest intensity. However, to get a better estimate of the PPG, a difference signal (x_δ) is obtained by subtracting the mean blue intensities from the green. The reason being, the similarity in the spectrum of green and blue channels in terms of the noise floor. Hence, the subtraction will result in isolation of

the higher intensity green signal minus the common noise.



(a) (b)

Figure 5.11: (a) Plot displaying the human skin absorptivity and camera sensitivity to RGB wavelengths of light and (b) Plot depicting variation in AC amplitude of PPG with wavelength of light.

The extracted PPG signal is then normalized by subtracting each data point by the mean and dividing it by the standard deviation of the signal. The normalized signal is filtered using the Daubechies04 wavelets to eliminate the high frequency noise, a method that has been shown to work well [66]. Since implementations on mobile devices are challenged by the reduced sampling rates compared to commercial biosensors, the number of scales of signal decomposition is estimated based on the sampling rate of the PPG obtained using the Android device. The frequency band of interest lies in the region from 1Hz - 4Hz which corresponds to 60 - 240 bpm, hence the appropriate scale of detail coefficients are eliminated in the reconstruction of the signal. A moving average of the filtered signal with a window size of five is computed to further smoothen the signal and amplify the peaks. A peak detection algorithm is then used to find the peaks in the signal by detecting the local maxima (Fig. 5.12). The number of peaks is used to compute the heart rate using (5.7).

$$HeartRate = \frac{No.ofPeaks * 60}{time} \tag{5.7}$$

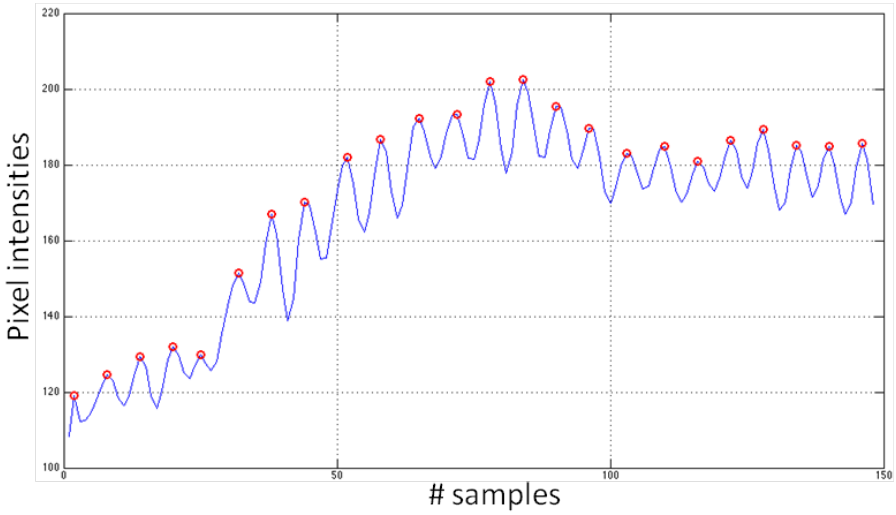


Figure 5.12: Plot depicting the PPG signal and its corresponding peak locations.

5.4.2 Oxygen Saturation

Oxygen saturation is defined as the ratio of the amount of oxygen bound with the hemoglobin molecules in the blood to the total oxygen capacity (5.8). The principle behind estimation of oxygen saturation (SpO2) in the arterial blood using the Photoplethysmogram (PPG) is derived from the theory of Beer-Lambert's law. The law states that the absorption of light by a medium causes the incident light intensity to drop exponentially, and the amount of absorption is related to the wavelength of light. Fig. **??** shows the absorption profiles of hemoglobin (Hb) and oxy-hemoglobin (HbO2) for a range of wavelengths between 200 nm and 1000 nm. Measuring blood gases is a crucial step to assess the level of oxygen saturation in a patient and to prevent hypoxic conditions. In a clinical setting, often times this requires invasive extraction of blood samples using arterial catheters. There is also a time lapse between the collection of the sample and its evaluation, and the results indicate status of the patient at the time of sampling. The need for a non-invasive and an instant method

of analyzing blood oxygen saturation will help prevent diagnostic errors especially in situations where rapid changes occur within a short span of time.

$$SpO_2\% = \frac{HbO_2}{Hb + HbO_2} \tag{5.8}$$

Arterial oxygen saturation refers to the amount of oxygen carried by hemoglobin in arterial blood. The principle idea behind measurement of oxygen saturation is the differential optical absorbance of oxyhemoglobin versus deoxyhemoglobin. From Fig. **??**, it can be observed that HbO_2 absorbs less light compared to Hb in the regions corresponding to the red wavelength. This is also why oxygenated blood looks distinctively red while de-oxygenated blood has a characteristic dark blue [67].

In this work, the red and blue channel data from the recorded video are used to measure SpO_2 in the arterial blood. These two channels are chosen based on the idea of selecting two light sources, one that amplifies the difference in absorption due to blood oxygenation and the other that is fairly invariant to this phenomenon. Here, the blue channel is used as a reference since the absorption of light is similar for both HbO2 and Hb. In addition, blood viscosity affects the flux of blood in the capillaries, which in turn affects the shape of the PPG signal. Two popular approaches to estimate SpO_2 involve fitting a linear equation [12], and using the molar extinction coefficients of Hb and HbO_2 [68].

An important step in the estimation of SpO_2 is the calculation of a normalized ratio (*Ratio*) of the red to the blue transmitted light. It is computed as the ratio of the AC components to DC components of each of the light sources (5.9). The AC components are represented by the standard deviation while the DC components are represented by the mean, and these values are estimated for every frame of the video. Each *Ratio* value can be used to compute the SpO_2 which implies that a new reading can be recorded every second. Given that the SpO_2 value cannot vary by more than

2% [69], a better representation of this *Ratio* is required to improve the accuracy of the SpO_2 estimation. The proposed algorithm to obtain SpO_2 from PPG is solved by fitting a regression model to a linear system of equations (5.10). A histogram of the *Ratio* is used as a feature to get a stabilized SpO_2 reading. Therefore, each $\mathbf{R}$ is an n-dimensional vector, where n is the number of bins the ratio values are put into (5.12).

$$Ratio = \frac{(I_{AC}/I_{DC})_{RED}}{(I_{AC}/I_{DC})_{BLUE}} \tag{5.9}$$

$$\mathbf{y} = \mathbf{M}\mathbf{a} + b\mathbf{1}, \tag{5.10}$$

where $\mathbf{y} \in \mathbb{R}^{n\times 1}$ denotes the ground truth SpO_2 values obtained for each subject using a commercial pulse oximeter, $\mathbf{a}$, b are the unknown parameters of the linear regression model, and $\mathbf{1}$ is the vector of ones. In order to estimate the unknown parameters, we solve the following optimization problem.

$$\min_{\mathbf{x}} \|\mathbf{y} - \mathbf{M}\mathbf{x}\|_2^2 + \lambda\|\mathbf{x}\|_2^2. \tag{5.11}$$

Here,

$$\mathbf{M} = \begin{bmatrix} \mathbf{R}_1 & \mathbf{R}_2 & \cdots & \mathbf{R}_n \\ 1 & 1 & \cdots & 1 \end{bmatrix} \tag{5.12}$$

$$\mathbf{x} = \begin{bmatrix} \mathbf{a} \\ b \end{bmatrix}$$

The solution to this ℓ_2 regularized (ridge regression) least squares problem can be obtained as

$$\mathbf{x} = (\mathbf{M}^T\mathbf{M} + \lambda\mathbb{I})^{-1}\mathbf{M}^T\mathbf{y}. \tag{5.13}$$

The regularization parameter λ is added to the optimization problem to stabilize the solution in case $\mathbf{M}^T\mathbf{M}$ are numerically ill-conditioned or misbehaving. In addition, it ensures that the model is not over-fitted to the training data which will otherwise lead to poor performances on the test data and a lack of algorithm generalization.

5.4.3 Experiments and Results

In this section, we provide details about the dataset used to evaluate our algorithm and present the heart rate and oxygen saturation estimation results. The results are compared to the values obtained using a commercial pulse oximeter that served as the ground truth measurement. The data collection involved recording a video of a subject's finger tip for 20 seconds using the first generation Android Nexus 7 and the Samsung Galaxy 2 tablets. Typically, these two devices provide a video recording with a sampling rate ranging from 15 to 30 frames per second.

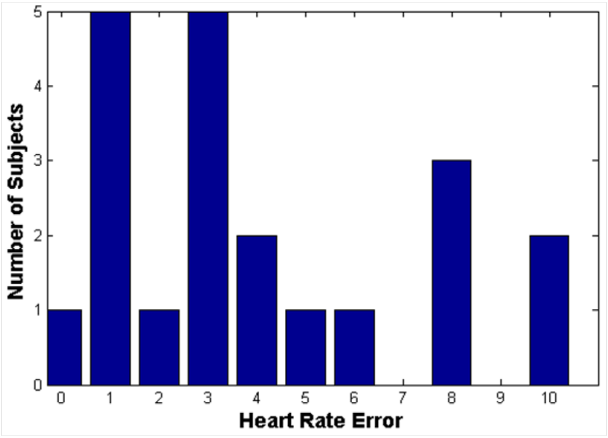


Figure 5.13: Histogram of error obtained in the estimation of heart rate from PPG using multiresolution analysis.

For the heart rate algorithm, three different recordings were measured from each subject (age 18-30) which comprised of a combination of different breathing patterns such as normal, fast, and deep breaths. The algorithm was tested against 21 subjects and an overall average error of about 5.84% was obtained (Fig. 5.13).

For the oxygen saturation algorithm, video recordings of the finger tip acquired from 6 subjects was used to train the model. The results obtained are provided in Table 5.1.

Table 5.1: Performance of the Proposed Approach to estimate SpO_2

Subject ID	**Grouth Truth (%)**	**Estimated (%)**
1	99	99.04
2	98	98.11
3	99	97.11
4	99	98.99
5	98	95.71
6	98	96.53

5.4.4 Android Interface for m-Health

The m-Health module for PPG extraction and parameter estimation is depicted in Fig. 5.14. To use the module, users must place their finger (index preferably) on the camera lens as still as possible and record a 20 second video. Once the timer has counted down to zero, the gauge meter displays the heart rate estimated. On performing a swipe action over the gauge, a dashboard with the parameter values, plot, and menu options can be visualized (Fig. 5.15). Start and stop buttons are provided to control the signal acquisition and additionally, there is a save button that allows for data storage on the sd card (Fig. 5.16).

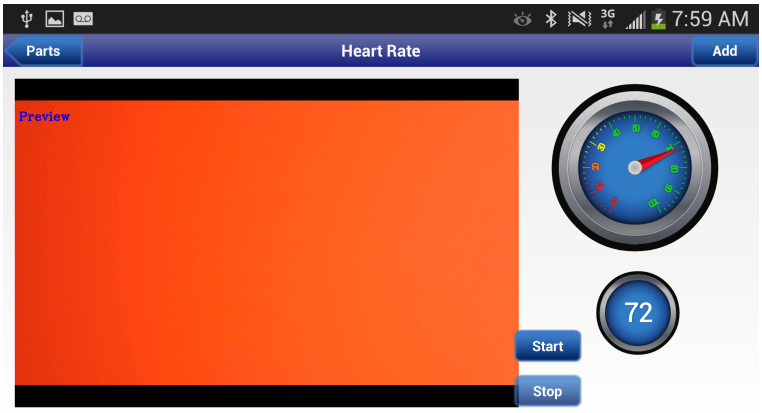


Figure 5.14: User interface of the PPG module displaying the acquisition of a video for 20 seconds.

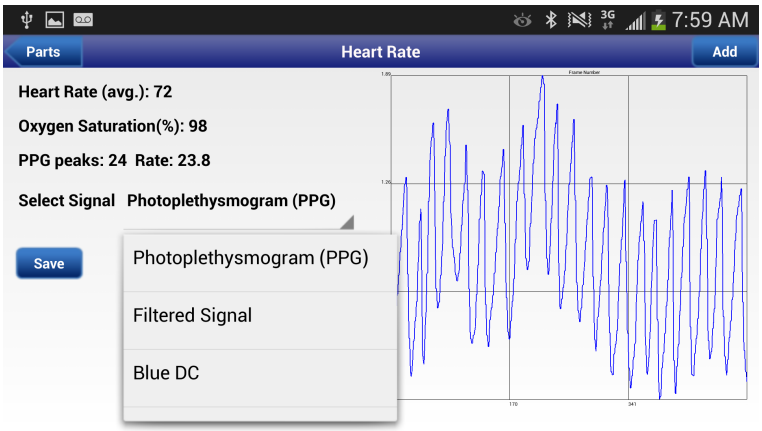


Figure 5.15: User interface to visualize the PPG signal and the estimated heart rate and oxygen saturation values. A menu option for selecting the different signals is also shown.

5.5 Step Counter using Accelerometer

The accelerometer can be used to measure instantaneous acceleration due to forces acting on the sensor in the X, Y and Z directions. Analysis of accelerometer data requires extraction of features such as mean, standard deviation, and energy for each axis, and correlations between axes. The accelerometer signal provides information on the activity of a person. In particular, human activity recognition opens the door to a world of healthcare applications such as elder care, cognitive assistance and fitness monitoring [14]. Most applications require data collection from the accelerometer

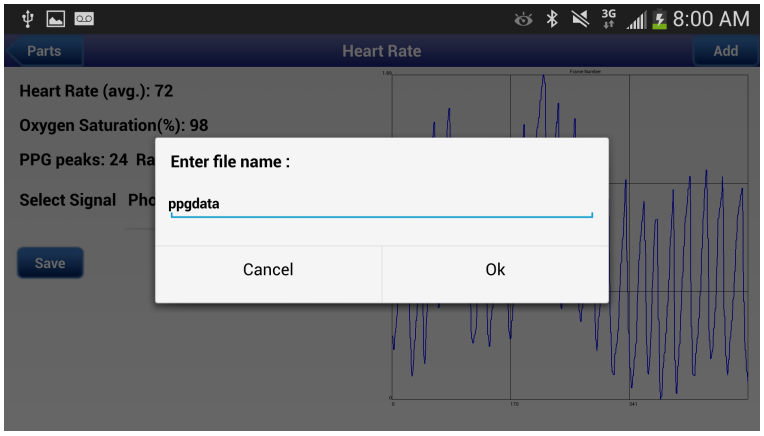


Figure 5.16: Dialog option to save the recorded PPG signal as a text file on the sd card of the device.

placed in multiple body locations to be able to monitor activity. However, using multimodal sensor data by combining the accelerometer with other sensors such as GPS, Gyroscope, Microphone, Light sensor and so on can be more effective in context-aware applications where the data collection can be reduced to a single location. For instance, having both GPS data and accelerometer signal a better estimate of the activity of driving can be obtained as against just having acceleration information. Modern mobile devices come with a built-in accelerometer that can be used to track simple activities such as walking, running, climbing stairs, biking etc.

5.5.1 Step Count Estimation

In this section, a description of the algorithm developed to count the number of steps measured using the built-in Android accelerometer is presented along with the results obtained and the Android interface developed.

The basic idea behind step counting is to detect the point at which the heel of the foot that is off the ground is placed back on the ground during an activity involving a step. This corresponds to the zero crossing of the accelerometer signal that measures acceleration caused due to the forces acting on it including the force of

gravity. The proposed algorithm was derived from that described in [70]. Fig. 5.17 gives an overview of the proposed algorithm to count steps. The raw accelerometer data is depicted in Fig 5.18 and it can be observed that the signal is corrupted with a significant amount of noise. This is one of the main challenges when dealing with using mobile devices as the signal acquisition platform in data driven applications. The key to building an accurate step counting algorithm requires a filter that eliminates high frequency while preserving signal shape.

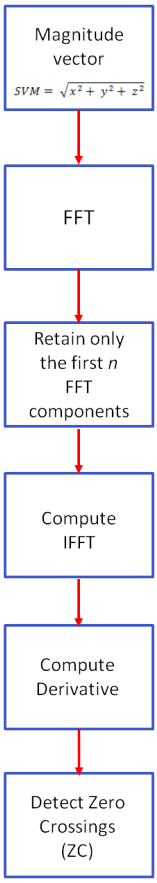


Figure 5.17: Block diagram illustrating the sequence of steps in the algorithm to estimate step count using Android accelerometer.

In the proposed algorithm, the magnitude of the raw accelerometer signal is first computed to eliminate dependencies on the orientation of the mobile device. The acceleration due to the force of gravity (taken as 9.8m/s) is then subtracted from the

magnitude vector, to generate a linear acceleration signal caused only due to forces acting on the 3-axis accelerometer. The signal magnitude vector (SVM) is then transformed using an N-point FFT, where N is the signal length. A low-pass filter is then applied to eliminate the high frequency noise by a process of picking only the first k frequency components. This parameter k is tuned for every subject to improve the accuracy of the step count estimation. The inverse transform (IFFT) is then computed and its real part is preserved. Fig. 5.19 depicts a smooth accelerometer signal obtained after the filtering stage, and the sinusoidal pattern observed in the signal corresponds to the necessary step information. A derivative of the signal is then computed using (5.14) to make the high-pass nature of the filtered signal more prominent to facilitate detection of the zero crossings. From Fig. 5.20, the increased prominence of the sinusoidal pattern that was embedded in the filtered signal can be visualized. A zero crossing detector is then applied to $y[n]$, where a zero crossing is defined as the point at which the signal crosses from a positive to a negative value. The number of zero crossings in the signal corresponds to the number of steps.

$$y[n] = 2x[n] + x[n-1] - x[n-3] - 2x[n-4] \tag{5.14}$$

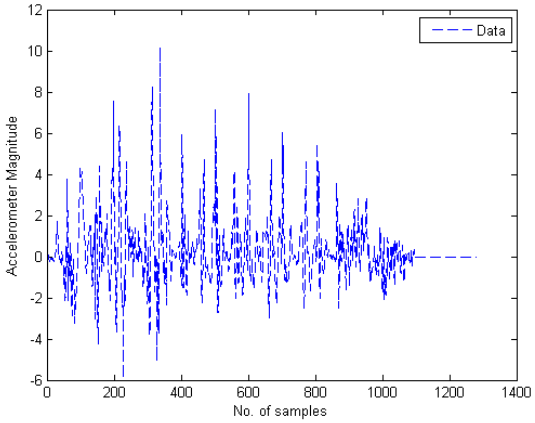


Figure 5.18: Raw signal magnitude obtained from the Android built-in accelerometer while a subject walked 10 steps.

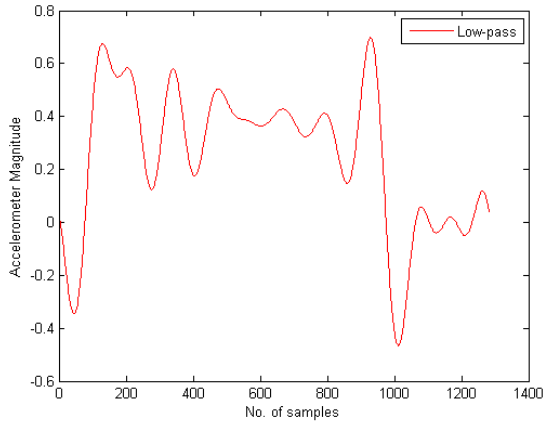


Figure 5.19: Plot displaying the accelerometer signal obtained after low-pass filtering.

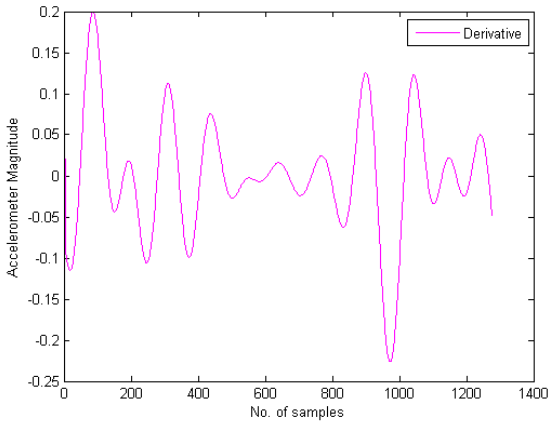


Figure 5.20: Plot depicting the derivative of the signal in Fig. 5.19 showing 10 prominent peaks corresponding to the 10 steps walked by the subject.

5.5.2 Experiments and Results

In this section, we provide details about the dataset used to evaluate our algorithm and present the step count estimation results. The results are compared to the manual measurement of the subject which serves as the ground truth.

The data collection involved using the developed Android interface to record regular walking and limping activities. The subjects were requested to place the mobile device in their pockets and perform a pre-determined activity that involves a step. Once the activity was completed, the devices were removed from the pocket

and a save button provided on the user interface was used to save the raw signal magnitude. The data was annotated by the subjects as they are asked to make a note of the number of steps they take. The algorithm was tested on a total of 26 subjects and an overall average error of 9% was obtained. The average error in the number of steps was only 0.615 and Fig. 5.21 shows the histogram of error obtained for every subject.

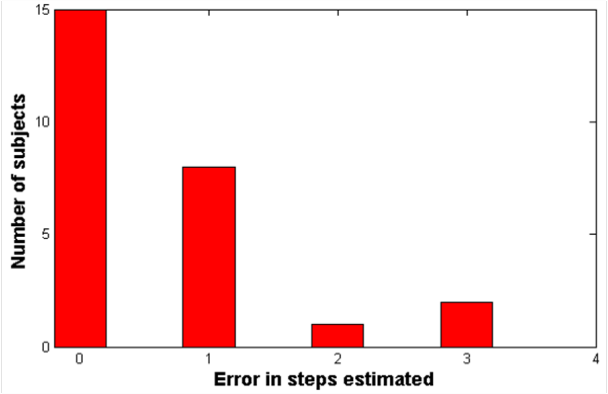


Figure 5.21: Plot depicting a histogram of the error obtained in the step count estimation on data comprising of walking and limp movements.

Figure 5.22: The Android user interface developed to demonstrate a step counter.

5.5.3 Android Interface for Step Counting

The step counter user interface in AJDSP is depicted in Fig. 5.22. Data is streamed from the built-in Android 3-axis accelerometer, and it can be controlled using the start and stop buttons. The instantaneous acceleration values for each of the X, Y and Z-axis are displayed along with the step count estimated. By performing a swipe action on the plot, the different frames of the signal can be visualized. The output of the block can be either of the three axis vectors or the signal magnitude vector which can be set using the pull-down menu. A parse button which is to transfer the collected data to other AJDSP modules for processing has also been provided in the user interface.

Chapter 6

LABORATORY EXERCISES

6.1 Introduction

In this section, a few sample laboratory exercises designed for undergraduate students belonging to the STEM program is described. Performing laboratories using these sensors provides an overview of the procedure of collecting sensor measurements from the different sensors, and the abilities of mobile devices to act as computational signal analysis platforms. Additionally, the AJDSP block functionality of processing signals in a frame-by-frame manner gives a better intuition of the processing operation that takes place, and a way to closely observe important signal characteristics. The interface between sensors and mobile apps can be used to present students with the diverse applications of DSP.

6.2 Denoising Speech Signals by FIR Band-stop Filter Design

The objective of this exercise is to demonstrate how to use a band-stop filter to remove an interfering signal from speech. For the simulation (Fig. 6.1), a Sound Recorder block is used to record a speech signal, to which a sinusoid is added to simulate narrow-band noise. The students are asked to listen to both the original speech signal and the noisy signal using the Sound Player block. Subsequently, they are instructed to process the resulting signal by a band-stop filter to remove the interfering sinusoid. The filter is designed using the FIR Design block by specifying the cut-off frequencies, the stop-band and pass-band attenuation characteristics, and window type. The output of the filter is listened to by the students and compared

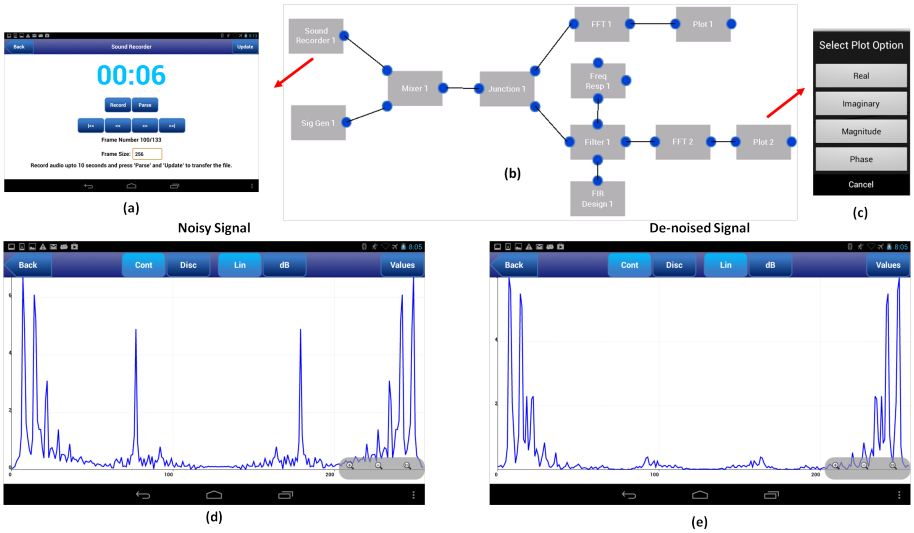


Figure 6.1: Simulation to perform noise removal from speech signals by designing a band-stop FIR filter. (a) Sound recorder user interface, (b) modules connected to set-up the simulation, (c) Dialog option in the plot module, (d) Input audio corrupted with the sinusoid (symmetric peaks), (e) Denoised output signal with the sinusoid suppressed (no peaks).

with the original signal to examine the effectiveness of the filter in removing the noise. Furthermore, the spectra of the noisy and de-noised signals are observed as shown in Fig. 6.1(d) and 6.1(e).

6.3 Understanding ECG Signals

Most physiological signals are non-stationary (frequency varies with time) and need to be observed in real-time to provide students a better perspective of its content. However, acquiring ECG signals from each student individually to work on an exercise can be cumbersome. Therefore, a few sample ECG recordings are provided internally within the ECG Signal Generator block. It also shows the buttons provided to enable frame-by-frame processing of the non-stationary signal. The objective of this exercise is to provide hands-on experiences with AJDSP and a basic understanding of ECG signal characteristics, parameter estimation, and filtering. The first part of the exercise provides an overview of various signal characteristics in relation to the cor-

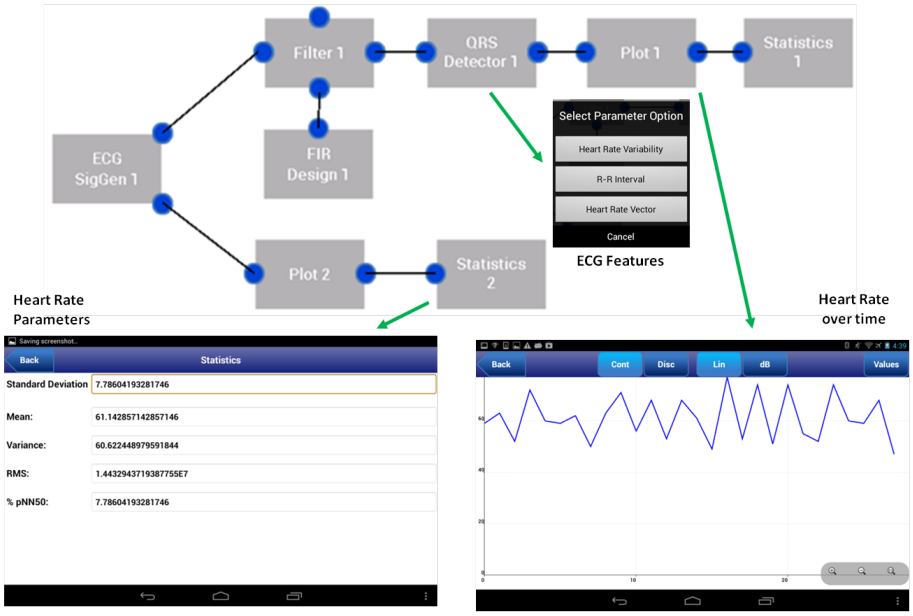


Figure 6.2: Simulation to understand characteristics of ECG and extract their features.

responding health conditions diagnosed from the patient data. Students are asked to use the ECG Signal Generator block to load the sample synthetic waveform obtained from Physionets ECGSYN toolkit [53]. They observe the P-wave, the QRS-complex and the T-wave of this artificially synthesized normal ECG and understand concepts of atrioventricular (AV) ratio, and the required ordering of the wave segments for a normal heart. The sampling rate of this signal and number of samples comprising the R-R time interval are provided to enable students to estimate the heart rate. Next, real ECG recordings, normal sinus rhythm, arterial fibrillation (AFB) and ventricular tachycardia (VT), and the corresponding waveforms are observed for these abnormalities. Furthermore, signals obtained using a lead VI configuration with and without baseline wandering (noise) are also provided and visualized [52].

6.4 ECG Feature Extraction

In the second part of the exercise, the baseline wandering artifact is removed by designing filters using the FIR Design block. The R-wave peaks are detected and relevant features are extracted using the QRS Detector block. The algorithm to detect these R-R peaks and extract features is based on multiresolution wavelet transform as described in [61]. The simulation diagram and expected outputs for this part of the exercise are shown in Fig. 6.2. Students can also visualize the frequency spectra of both the noisy and the de-noised ECG signals using the FFT block. Heart rate variability (HRV) analysis is conducted and feature vectors such as difference in successive R-R interval, mean and standard deviation of heart rate and R-R interval are calculated in the time-domain using the Statistics block.

6.5 Denoising ECG using Wavelets

The objective of this exercise is to use the property of multiresolution analysis to remove the effects of the high frequency power line interference in ECG. The number of scales of signal decomposition and the appropriate choice of wavelet are estimated. The simulation is set-up as shown in Fig. 6.3. The input ECG is set to be the noisy Lead VI configured signal. The Daubechies6 wavelet family is set in both the DWT and IDWT modules at first. The number of scales are varied and for each value, the output signal is observed using the Plot module. The output from the DWT module can be set to be equal to either the scaling coefficients (low-pass), the wavelet coefficients (high-pass), or the entire wavelet transformed signal which includes a stack of both scaling and wavelet coefficients. By setting the appropriate option for module output, a reasonable denoising operation is simulated. Next, the wavelet type parameter is varied and students can get a perspective on what families

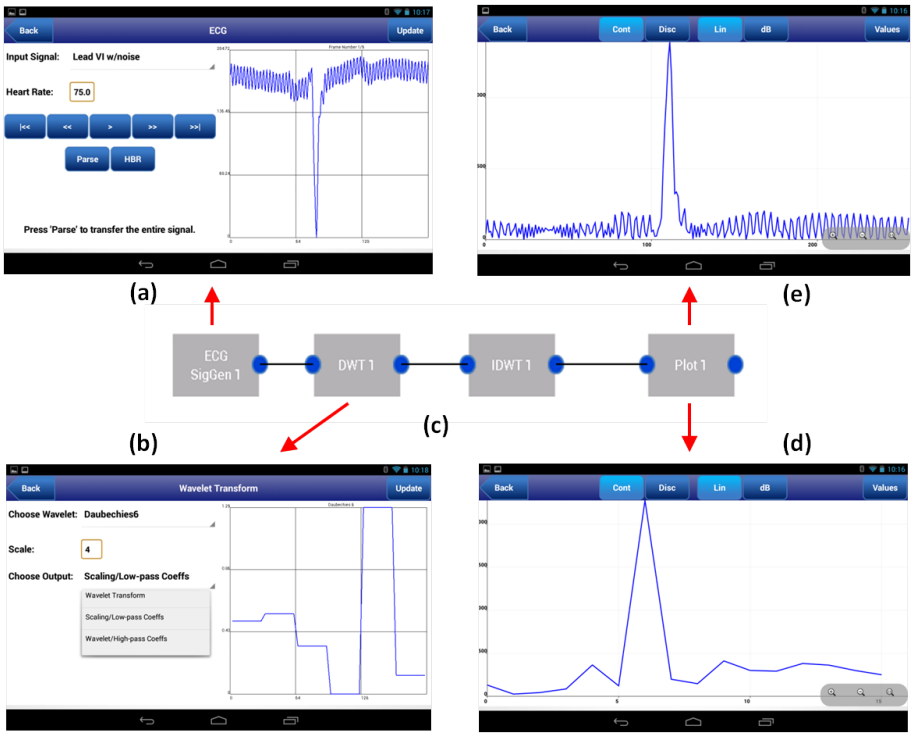


Figure 6.3: AJDSP simulation to denoise ECG using wavelet transform. (a) Biosignal generator displaying a noise Leavd VI ECG, (b) User interface of the DWT module, (c) Block-diagram of the simulation, (d) Denoised output signal, (e) Noisy input signal.

of wavelets work well when dealing with ECG applications. The noisy and denoised signals can be observed in the illustration provided.

6.6 Step Counting with Accelerometer

The objective of this exercise is to demonstrate a wireless DSP sensor system, to understand remote data acquisition, and to learn simple concepts about accelerometers and their role in context aware applications. In the first part of this exercise students are asked to use the Accelerometer block and select the option to connect to Shimmer sensors. The sensors are either held in their hand or strapped around their waist/ankle. They then establish a Bluetooth connection between the mobile device and the sensor through the app user interface. Data is streamed from the ac-

celerometer and the signal from each of the 3 axes is observed. The change in the axis experiencing the gravitational-force and the corresponding signal transitions based on different orientations of the sensor and activity of the subject is noted. In the second part of the exercise, students stream data from the accelerometer built-in-to the mobile device. They are asked to simultaneously stream data and calibrate the step counter by performing a predetermined activity, such as walking for five steps. Once calibrated, they then perform one of three activities: standing, walking and running. On stopping the data streaming, the total number of steps taken and the duration of each of the listed activities are displayed. By observing the frames of the signal, students learn that each step corresponds to a peak in the accelerometer magnitude and sharp signal transitions mark a change in activity.

6.7 PPG Extraction and Bio-parameter Estimation using a Camera

The objective of this exercise is to demonstrate a non-invasive health monitoring system by using the camera to extract a physiological signal. In the first part of this exercise students record a video of their finger tip using the Health Meter block. A preview of the video is observed during the recording to understand the optical principle behind the PPG signal. A pulsating light ring within the preview can be visualized with care to understand the physiological property of the signal obtained as every pulsing ring corresponds to a heartbeat [64]. After about 15 seconds, the recording stops and a meter displays the estimated heart rate. The extracted PPG waveform can be observed along with parameters such as oxygen saturation and respiratory rate. In the second part of the exercise students learn about extracting the frequency band that corresponds to the respiratory information in the PPG signal using the Wavelet Transform and Inverse Wavelet Transform blocks. The different bands can be observed by selecting the low frequency detail coefficients to be pre-

served during reconstruction of the transformed signal. Concepts on the relationship between scale, wavelet width and frequency can be understood using the wavelet blocks. The significance of approximation and detail coefficients and their relationship to signal energy can also be understood. In addition, a brief idea on the different family of wavelets that exist and choosing the right one in relevance to the application is provided.

Chapter 7

ASSESSMENTS, RESULTS AND OUTREACH

The AJDSP app was assessed through interactive workshops involving graduate and undergraduate students at Arizona State University. A detailed evaluation methodology was designed to test various aspects of the app such as: educational value, robustness, and improvement in conceptual understanding.

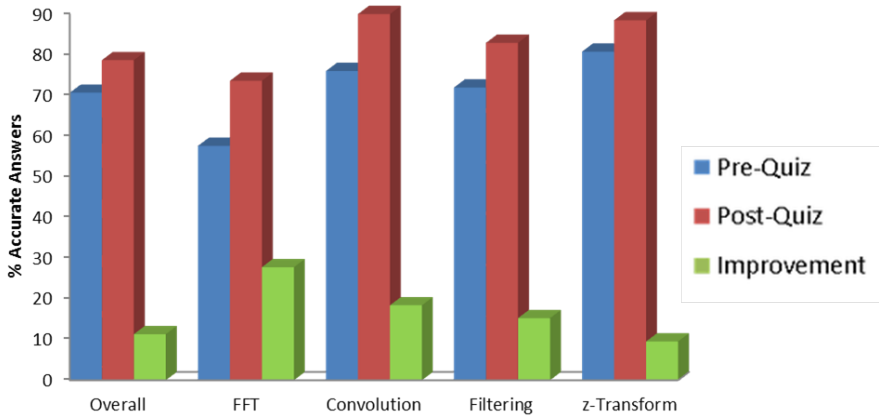


Figure 7.1: Improvement shown in student understanding of different DSP concepts.

The first stage of evaluation included a round of preliminary assessments based on the basic signal processing modules and their usefulness in reinforcing DSP fundamentals. These assessments were geared towards evaluating students' responses on using a graphical programming tool to perform laboratories as part of a DSP curriculum, and it was conducted in the Spring 2013 semester. In addition, the app was tested to find bugs and, feedback on the user interface, interactivity, speed, and ease-of-use were collected to help improve the conceptual design of the proposed software. A total of thirty-three students participated in the assessment workshops, and the responses provided by them were positive. The results showed a 11% improvement in

the overall understanding of DSP concepts, among topics such as: filter design, the FFT, z-transforms, and convolution (Fig. 7.1).

More recently, detailed assessments were conducted in the Fall 2013 semester, where two workshops were held, one in the Freshmen Engineering course (FSE 100) and the other with graduate students who have a DSP background (EEE 407/591). These assessments were geared towards evaluating the proposed m-Health modules and their effects in introducing DSP principles in health monitoring applications. The mobile healthcare domain was exploited to provide students with hands-on experiences in mobile sensing, processing, parameter estimation, and visualization. The following sections will discuss the individual workshops and their outcomes.

7.1 Freshmen Engineering Course Workshop

Arizona State University created a specialized department for freshmen engineering a few years ago to stimulate student interest in engineering. The program is customized to help students figure out their passion by providing an overview of several engineering domains through a perfect balance of lecture and laboratory sessions. For instance, in this semester there was an electrical lab, an aerodynamics lab, an engineering design process lab, and a matlab and newton's law of cooling lab. In addition, for the first time, a signal processing lab was conducted using the AJDSP app. In this course, students worked in groups of four and a total of 31 students participated in the assessment workshop. To perform the laboratory exercises, the Samsung Galaxy tablets were used, and each group was provided with two tablets.

The assessment workshop comprises of three components, namely: a pre-lab quiz, a hands-on simulation exercise, and a post-lab quiz. Students were first administered a pre-lab quiz that consisted of 13 multiple choice questions based on fundamentals of DSP and physiological signals. The quiz was followed by an interac-

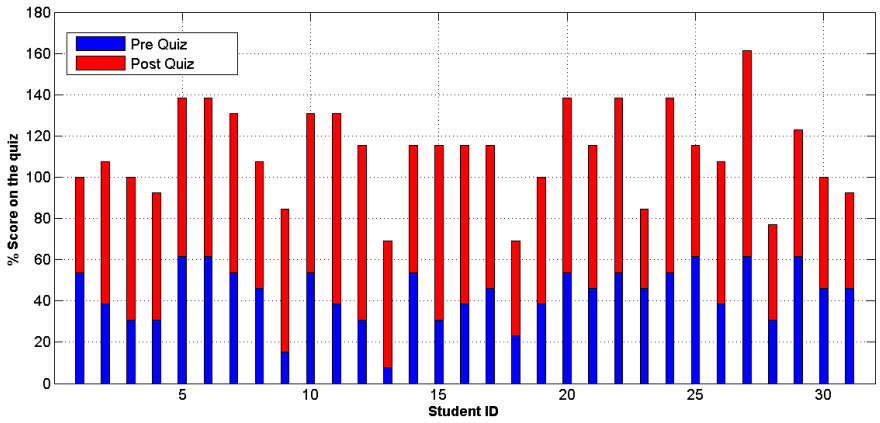


Figure 7.2: Performance of undergraduate FSE 100 students 35 in total, 4 girls and 31 boys.

tive presentation introducing them to signals, sensors and their applications particularly in health monitoring. The DSP exercises included understanding ECG signals, denoising speech signals using FIR filters, a step counter application using an accelerometer, and physiological parameter estimation using the photoplethysmogram signal. These exercises have been described in chapter 6 and a detailed step by step instruction of performing the simulations have been given in appendix A. The workshop also assisted in the refinement of algorithms through the process of sensor data collection. The students dabbled with several modules in AJDSP and were inquisitive about the science behind their operation. A post-lab questionaire was later uploaded on Blackboard which consisted of general evaluation, feedback and conceptual essay questions in addition to the pre-lab questions.

The freshmen students reported an overall improvement of 87.9186% in test scores. The overall performance improvement in the post-quiz compared to the pre-quiz is illustrated in Fig. 7.2. In addition, a normalized average gain also known as Hake gain of 43.61% was obtained. It is computed using (7.1), where $< pre\% >$ and $< post\% >$ are the average scores in the pre and post tests respectively.

$$HakeGain = \frac{(< post\% > - < pre\% >)}{(100 - < pre\% >)} \tag{7.1}$$

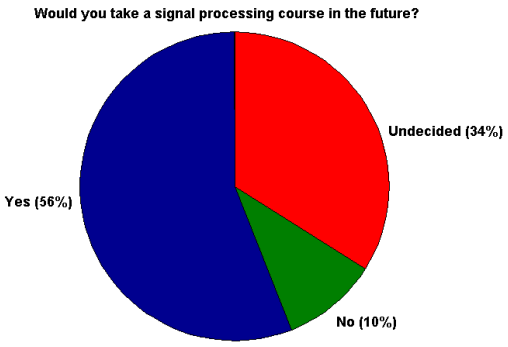


Figure 7.3: Chart indicating response of students when asked if they would consider taking a signal processing course in the future.

Responses from the students on the AJDSP app were positive and when asked if they would consider taking a signal processing course in the future, about 56% gave a positive response, with 34% being undecided (Fig. 7.3). Students showed a large improvement in understanding concepts such as applications of PPG signals, accelerometers, characteristics of ECG, and oxygen saturation. While, concepts on signal dimensionality, stationarity with respect to its frequency content, estimating heart rate, and the FFT showed slight or no improvement as seen in Fig. 7.4. This is expected given their lack of signal processing background and given that these are in general more involved concepts requiring higher mathematics skills. In addition, about 78% of the students said they took less than 10 minutes to get used to the simulation environment (Fig. 7.5). The results indicate that AJDSP has been successful in giving an application-based perspective to the abstract field of signal processing through its sensor interfaces and m-Health modules.

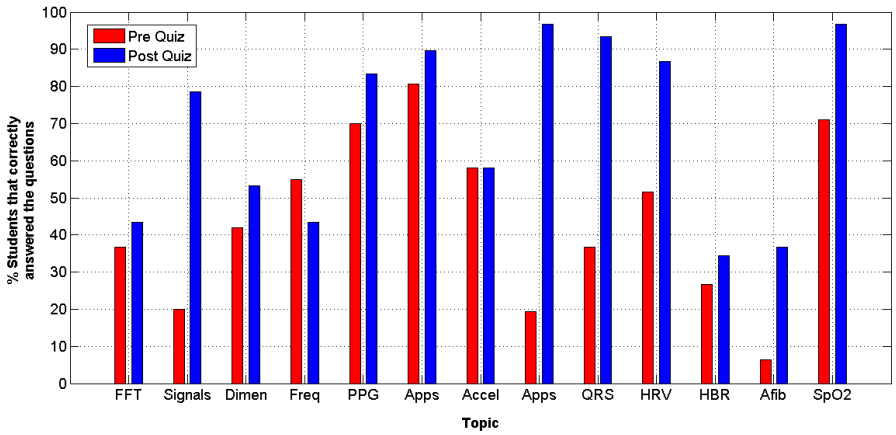


Figure 7.4: Topic-wise % improvement of students from the FSE 100 class.

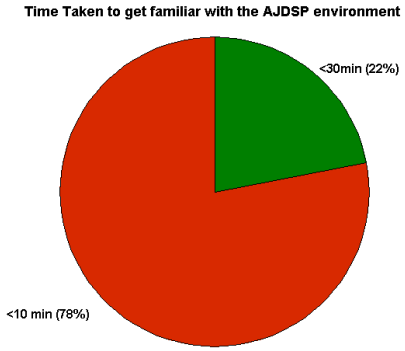


Figure 7.5: Chart depicting student response on the time take to get used to the simulation environment in AJDSP.

7.2 Graduate DSP Student Workshop

To further assess the AJDSP app, a second workshop was conducted involving graduate students with a DSP background. A control group experiment was performed where the students were divided into two groups, one that was administered the post-quiz immediately after the presentation, while the second group performed simulation exercises on AJDSP prior to being given the post-quiz. A total of 14 students participated in the workshop including 2 girls and 12 boys, and each group comprised of 7 students. The evaluation procedure followed a similar pattern to

that used in the FSE course. However, the pre-lab quiz comprised of 17 questions and, the simulation exercises included denoising ECG signals using wavelets, understanding ECG signals, heart rate estimation using PPG and data collection using the accelerometer. Fig. 7.6 depicts the average test scores obtained by the two student groups in both the pre and post quizzes. It can be observed that students belonging to group 2 show a greater improvement in performance having used the AJDSP app to run simulations in addition to the lecture presentation. The overall improvement in the test scores reported by group 1 (presentation only) was 19.33% as compared to the 37.82% reported by group 2 (presentation + AJDSP exercises). An illustration of the individual performances of each student belonging to both groups is provided in Fig. 7.7. In addition, the Hake gain for group 1 was 35.92% and group 2 was 59.23%. These results evidence that using the AJDSP app helps learn DSP concepts demonstrated through mobile health monitoring applications in a more thorough fashion.

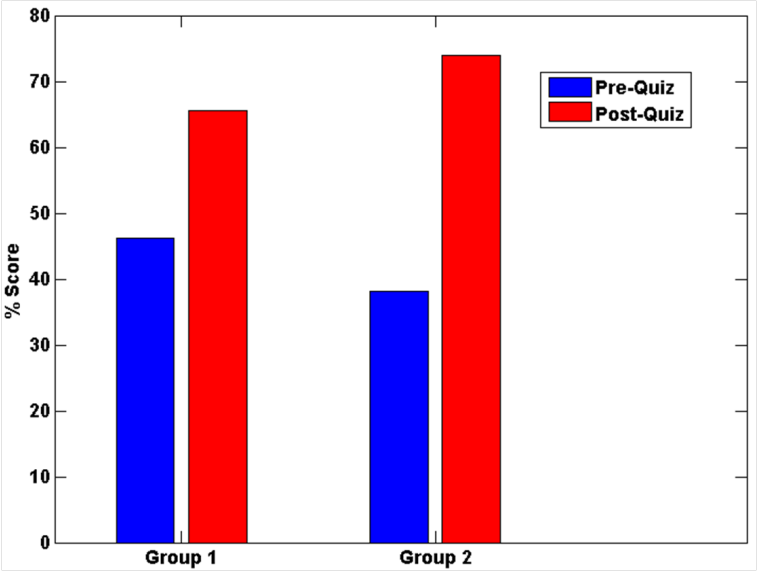


Figure 7.6: Average test scores obtained by graduate DSP students in the pre and post quizzes.

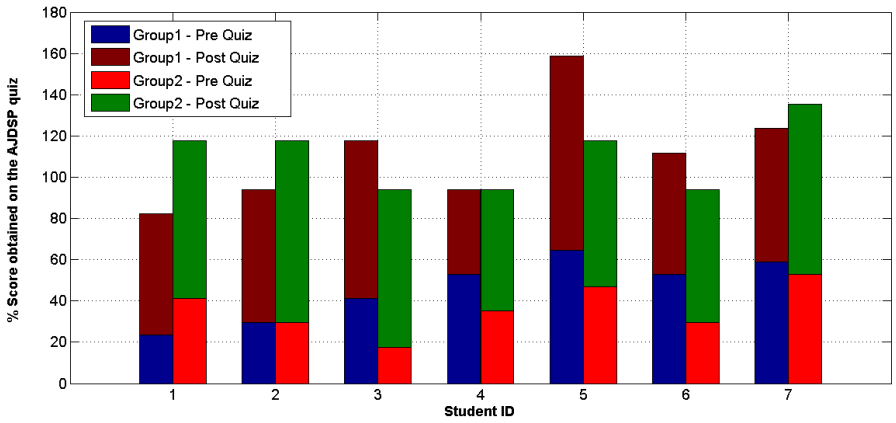


Figure 7.7: Performance of graduate DSP students in the control group experiment.

7.3 Outreach

As part of outreach, the AJDSP app was displayed at the Engineering Open House event conducted at ASU. The event was designed to showcase engineering and technology, in a creative and interactive fashion to students from K through 8th Grade. In addition, a demonstration of the AJDSP app and its various utilities was given at the Corona Del Sol High School, Tempe. Real-time streaming of data from the different sensors was one of the key attractive features as per the response from the students. To conclude, most of the students always showed a large amount of enthusiasm in the app and expressed their desire to install it on their personal Android devices.

Chapter 8

CONCLUSIONS

This work presents the interactive software modules developed for the Android platform that is capable of interfacing to various sensors, both on-board and external. The modules were utilized to demonstrate potential non-invasive health monitoring applications using mobile devices. Furthermore, the use of these modules in aiding DSP education was explored by designing creative laboratory exercises to provide students with hands-on experiences of data driven applications. The new software modules in the AJDSP app help to serve as a platform to visualize physiological signals such as PPG, ECG and GSR. In addition, modules to extract features and compute parameters such as heart rate, oxygen saturation and step count were developed. The power of modern mobiles and the feasibility of mobile healthcare (m-Health) have also been evidenced by the performance of the proposed algorithms. The ability to provide a way for easy access to vital health information and sensor data through the compact and ubiquitous mobile devices without the need for expensive sensor hardware can make a huge impact in healthcare. This opens up a large number of opportunities for innovation in signal processing research and education.

The software modules were used to create laboratory exercises, samples of which were provided. Several interactive workshops were conducted to evaluate the app and statistics were computed based on the data collected. The assessment results show promise in improving students understanding of DSP concepts when using the AJDSP app as part of the curriculum. It was observed that the ability to experience how sensors measurements are acquired and signals are processed helped students

better relate textbook knowledge to applications and creatively conceptualize their understanding. By depicting the challenges faced in the healthcare domain, students can be guided to innovate in these areas. In particular, the role of DSP in tackling problems in such real-world contexts can be presented. The AJDSP app with its unique combination of sophisticated DSP functionalities, the proposed sensing features, and an interactive workspace for performing block-based simulations make it a novel and valuable educational tool.

8.1 Implementation Challenges

There are still several challenges to be tackled in the implementation of data processing algorithms on mobile platforms. An important operating system (OS) requirement for continuous sensing is the capability of multi-tasking and background processing to enable constant streaming of sensor measurements. Therefore, while developing apps one must conform to these predefined constraints on access to resources. However, none of the OS profiles provide the ability to continuously stream data from all the sensors [4]. Mobile phones are used in varied contexts in day-to-day activities with a dynamically changing background environment. This especially creates unexpected conditions and scenarios for testing algorithms developed using machine learning techniques. Moreover, statistical models may also tend to fail to generalize under such environments.

Fragmentation is another reason to cause Android developers some inconvenience. Open Signal, a UK-based app developing company, has spotted 11,868 distinct Android devices over the last few months. In addition, the Android ecosystem is fragmented by OS, namely: Jelly Bean, Ice Cream Sandwich, and Gingerbread. Ensuring that the developed app is compatible with all device types running different OS versions can be quite exhausting.

8.2 Future Work

In addition to the existing features, AJDSP can be used to build context-aware applications such as stress detection, indoor navigation, and activity recognition. It can also be used to estimate Blood Pressure (BP) by calculating the Pulse Transit Time (PTT) and, detect arrhythmia and classify ECG signals into normal and abnormal beats. Other directions that can be looked into involve incorporating a way to connect sensors to the server and transmit data to the cloud to establish communication over the network.

Moreover, the capabilities of the AJDSP app can be extended to serve as a platform for scientific visualization by building sophisticated user interfaces that allow for graphical representation of complex data. These interfaces can be used to understand spatial and temporal relationships between the data to gain an insight in it. An application domain that can benefit enormously from such visualization capabilities is medical imaging.

REFERENCES

[1] M. Weiser and J. S. Brown, "The coming age of calm technology," in *Beyond calculation*. Springer, 1997, pp. 75–85.

[2] "Wearable computing image," Available online at http://www.fool.com/investing/general/2012/08/02/investing-in-the-wearable-computing-revolution-for.aspx.

[3] "Mobile cloud image," Available online at http://www.ibm.com/smarterplanet/us/en/smarter-enterprise/case-studies/us-open.html?cmp=usbrb&cm=j&csr=agus_brspuso-20130808&cr=vanity&ct=usbrb301&cn=usopen_vanity.

[4] N. D. Lane, E. Miluzzo, H. Lu, D. Peebles, T. Choudhury, and A. T. Campbell, "A survey of mobile phone sensing," *Communications Magazine, IEEE*, vol. 48, no. 9, pp. 140–150, 2010.

[5] I. Darwin, *Android Cookbook*. O'Reilly, 2012.

[6] "Sci at university of utah," Available at http://www.sci.utah.edu/jbresearch/resear-vis.html.

[7] "Emcl research group at hiedelberg university," Available at http://emcl.uni-hd.de/research_sv.html.

[8] P. Leijdekkers and V. Gay, "Personal heart monitoring system using smart phones to detect life threatening arrhythmias," in *Computer-Based Medical Systems, 2006. CBMS 2006. 19th IEEE International Symposium on*. IEEE, 2006, pp. 157–164.

[9] A. Homs-Corbera, J. A. Fiz, J. Morera, and R. Jané, "Time-frequency detection and analysis of wheezes during forced exhalation," *Biomedical Engineering, IEEE Transactions on*, vol. 51, no. 1, pp. 182–186, 2004.

[10] M. Rofouei, M. Sinclair, R. Bittner, T. Blank, N. Saw, G. DeJean, and J. Heffron, "A non-invasive wearable neck-cuff system for real-time sleep monitoring," in *Body Sensor Networks (BSN), 2011 International Conference on*. IEEE, 2011, pp. 156–161.

[11] J. A. Healey and R. W. Picard, "Detecting stress during real-world driving tasks using physiological sensors," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 6, no. 2, pp. 156–166, 2005.

[12] C. G. Scully, J. Lee, J. Meyer, A. M. Gorbach, D. Granquist-Fraser, Y. Mendelson, and K. H. Chon, "Physiological parameter monitoring from optical recordings with a mobile phone," *Biomedical Engineering, IEEE Transactions on*, vol. 59, no. 2, pp. 303–306, 2012.

[13] M.-Z. Poh, K. Kim, A. D. Goessling, N. C. Swenson, and R. W. Picard, "Heartphones: Sensor earphones and mobile application for non-obtrusive health monitoring," in *Wearable Computers, 2009. ISWC'09. International Symposium on*. IEEE, 2009, pp. 153–154.

[14] T. Choudhury, S. Consolvo, B. Harrison, J. Hightower, A. LaMarca, L. LeGrand, A. Rahimi, A. Rea, G. Bordello, B. Hemingway *et al.*, "The mobile sensing platform: An embedded activity recognition system," *Pervasive Computing, IEEE*, vol. 7, no. 2, pp. 32–41, 2008.

[15] N. D. Lane, M. Mohammod, M. Lin, X. Yang, H. Lu, S. Ali, A. Doryab, E. Berke, T. Choudhury, and A. Campbell, "Bewell: A smartphone application to monitor, model and promote wellbeing," in *5th International Conference on Pervasive Computing Technologies for Healthcare (PervasiveHealth2011)*, 2011.

[16] R.-G. Lee, K.-C. Chen, C.-C. Hsiao, and C.-L. Tseng, "A mobile care system with alert mechanism," *Information Technology in Biomedicine, IEEE Transactions on*, vol. 11, no. 5, pp. 507–517, 2007.

[17] E. Miluzzo, N. D. Lane, K. Fodor, R. Peterson, H. Lu, M. Musolesi, S. B. Eisenman, X. Zheng, and A. T. Campbell, "Sensing meets mobile social networks: the design, implementation and evaluation of the cenceme application," in *Proceedings of the 6th ACM conference on Embedded network sensor systems*. ACM, 2008, pp. 337–350.

[18] "Mobisante," Available online at http://www.mobisante.com/.

[19] "Fitbit android app," Available online at http://www.fitbit.com/android.

[20] "Ginger.io," Available online at http://ginger.io/.

[21] "Apps image," Available online at http://gettingsmart.com/2013/03/infographic-k-12-technology-usage/.

[22] "Endomondo," Available online at http://www.endomondo.com/.

[23] "Instant heart rate pro and stress check," Available online at http://www.azumio.com/apps/.

[24] "Blood pressure journal," Available online at https://play.google.com/store/apps/details?id=com.michaelfester.heart &hl=en.

[25] "Blackboard mobile," Available online at http://www.blackboard.com/platforms/mobile/overview.aspx.

[26] "Class dojo," Available online at http://www.classdojo.com/.

[27] "The calculus tools," Available online at https://play.google.com/store/apps/details?id=com.andymc.derivative &hl=en.

[28] "The cram," Available online at http://mac.appstorm.net/reviews/cram-the-mac-studyassistant/.

[29] "Matlab mobile," Available online at http://www.mathworks.com/mobile/ index.html/.

[30] "Wolfram alpha," Available online at http://products.wolframalpha.com/android/.

[31] "Octave," Available online athttps://play.google.com/store/apps/ details?id=com.octave&hl=en.

[32] J. Liu, S. Hu, J. J. Thiagarajan, X. Zhang, S. Ranganath, M. K. Banavar, and A. Spanias, "Interactive dsp laboratories on mobile phones and tablets," in *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*. IEEE, 2012, pp. 2761–2764.

[33] J. Liu, A. Spanias, M. K. Banavar, J. J. Thiagarajan, K. N. Ramamurthy, S. Hu, and X. Zhang, "Work in progressinteractive signal-processing labs and simulations on ios devices," in *Frontiers in Education Conference (FIE), 2011*. IEEE, 2011, pp. F2G–1.

[34] S. Ranganath, J. J. Thiagarajan, K. N. Ramamurthy, S. Hu, M. Banavar, and A. Spanias, "Work in progress: Performing signal analysis laboratories using android devices," in *2012 Frontiers in Education Conference Proceedings*. IEEE, 2012, pp. 1–2.

[35] "ijdsp on itunes app store," Available online at https://itunes.apple.com/us/app/ijdsp/id540962535?mt=8.

[36] "The ajdsp app on google play," Available online at https://play.google.com/store/apps/details?id=com.prototype.ajdsp&hl=en.

[37] Y. Rogers, "Moving on from weisers vision of calm computing: Engaging ubicomp experiences," in *UbiComp 2006: Ubiquitous Computing*. Springer, 2006, pp. 404–421.

[38] A. Spanias and V. Atti, "Interactive online undergraduate laboratories using j-dsp," *Education, IEEE Transactions on*, vol. 48, no. 4, pp. 735–749, 2005.

[39] H. Kwon, V. Berisha, V. Atti, and A. Spanias, "Experiments with sensor motes and java-dsp," *Education, IEEE Transactions on*, vol. 52, no. 2, pp. 257–262, 2009.

[40] L. Naismith, M. Sharples, G. Vavoula, P. Lonsdale *et al.*, "Literature review in mobile technologies and learning," 2004.

[41] J. Roschelle, M. Sharples, T.-W. Chan *et al.*, "Introduction to the special issue on wireless and mobile technologies in education." *J. Comp. Assisted Learning*, vol. 21, no. 3, pp. 159–161, 2005.

[42] C. Cortez, M. Nussbaum, X. López, P. Rodríguez, R. Santelices, R. Rosas, and V. Marianov, "Teachers' support with ad-hoc collaborative networks," *Journal of Computer Assisted Learning*, vol. 21, no. 3, pp. 171–180, 2005.

[43] K. M. Ahmed, "Wireless body area sensor networks signal processing and communication framework: Survey on sensing, communication technologies, delivery and feedback," *Journal of Computer Science*, vol. 8.

[44] "Mobisante," Available online at http://www.shimmer-research.com/products-2.

[45] A. Burns, B. R. Greene, M. J. McGrath, T. J. O'Shea, B. Kuris, S. M. Ayer, F. Stroiescu, and V. Cionca, "Shimmer–a wireless sensor platform for noninvasive biomedical research," *Sensors Journal, IEEE*, vol. 10, no. 9, pp. 1527–1534, 2010.

[46] E. Fortune, M. Tierney, C. N. Scanaill, A. Bourke, N. Kennedy, and J. Nelson, "Activity level classification algorithm using shimmer wearable sensors for individuals with rheumatoid arthritis," in *Engineering in Medicine and Biology Society, EMBC, 2011 Annual International Conference of the IEEE*. IEEE, 2011, pp. 3059–3062.

[47] F. Rincon, N. Boichat, D. Atienza, and N. Khaled, "Wavelet-based ecg delineation on a wearable embedded sensor platform," in *Wearable and Implantable Body Sensor Networks, 2009. BSN 2009. Sixth International Workshop on*. IEEE, 2009, pp. 256–261.

[48] K. Kanoun, H. Mamaghanian, N. Khaled, and D. Atienza, "A real-time compressed sensing-based personal electrocardiogram monitoring system," in *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2011*. IEEE, 2011, pp. 1–6.

[49] K. Lorincz, B.-r. Chen, G. W. Challen, A. R. Chowdhury, S. Patel, P. Bonato, M. Welsh *et al.*, "Mercury: a wearable sensor network platform for high-fidelity motion analysis." in *SenSys*, vol. 9, 2009, pp. 183–196.

[50] F.-T. Sun, C. Kuo, H.-T. Cheng, S. Buthpitiya, P. Collins, and M. Griss, "Activity-aware mental stress detection using physiological sensors," in *Mobile Computing, Applications, and Services*. Springer, 2012, pp. 211–230.

[51] D. O. Olguun and A. S. Pentland, "Human activity recognition: Accuracy across common locations for wearable sensors," 2006.

[52] "Ecg recordings," Available online at http://ocw.mit.edu/courses/healthsciences-and-technology/hst-582j-biomedical-signal-and-imageprocessing-spring-2007/labs/, http://courses.washington.edu/bioen303/Lab/labdatafiles2.htm#ONE.

[53] "Physionet ecgsyn toolkit," Available online at http://www.physionet.org/physiotools/ecgsyn/.

[54] "Wavelet tutorial," Available at http://users.rowan.edu/ polikar/WAVELETS/WTpart2.html.

[55] S. Mallat, *A wavelet tour of signal processing*. Access Online via Elsevier, 1999.

[56] P. S. Addison, "Wavelet transforms and the ecg: a review," *Physiological measurement*, vol. 26, no. 5, p. R155, 2005.

[57] A. Jensen and A. la Cour-Harbo, *Ripples in mathematics: the discrete wavelet transform*. springer, 2001.

[58] "jwave," Available at https://code.google.com/p/jwave/.

[59] B.-U. Kohler, C. Hennig, and R. Orglmeister, "The principles of software qrs detection," *Engineering in Medicine and Biology Magazine, IEEE*, vol. 21, no. 1, pp. 42–57, 2002.

[60] J. Mietus, C. Peng, I. Henry, R. Goldsmith, and A. Goldberger, "The pnnx files: re-examining a widely used heart rate variability measure," *Heart*, vol. 88, no. 4, pp. 378–380, 2002.

[61] S. Mahmoodabadi, A. Ahmadian, M. Abolhasani, M. Eslami, and J. Bidgoli, "Ecg feature extraction based on multiresolution wavelet transform," in *Engineering in Medicine and Biology Society, 2005. IEEE-EMBS 2005. 27th Annual International Conference of the*. IEEE, 2006, pp. 3902–3905.

[62] S. Banerjee, R. Gupta, and M. Mitra, "Delineation of ecg characteristic features using multiresolution wavelet analysis method," *Measurement*, vol. 45, no. 3, pp. 474–487, 2012.

[63] K. H. Shelley, "Photoplethysmography: beyond the calculation of arterial oxygen saturation and heart rate," *Anesthesia & Analgesia*, vol. 105, no. 6S Suppl, pp. S31–S36, 2007.

[64] P. Pelegris, K. Banitsas, T. Orbach, and K. Marias, "A novel method to detect heart beat rate using a mobile phone," in *Engineering in Medicine and Biology Society (EMBC), 2010 Annual International Conference of the IEEE*. IEEE, 2010, pp. 5488–5491.

[65] P. Sahindrakar, G. de Haan, and I. Kirenko, "Improving motion robustness of contact-less monitoring of heart rate using video analysis," *Technische Universiteit Eindhoven, Department of Mathematics and Computer Science*, 2011.

[66] T.-H. Fu, S.-H. Liu, and K.-T. Tang, "Heart rate extraction from photoplethysmogram waveform using wavelet multi-resolution analysis," *Journal of Medical and Biological Engineering*, vol. 28, no. 4, pp. 229–232, 2008.

[67] Y. Mendelson, "Pulse oximetry: theory and applications for noninvasive monitoring." *Clinical chemistry*, vol. 38, no. 9, pp. 1601–1607, 1992.

[68] G. M. Azmal, A. Al-Jumaily, and M. Al-Jaafreh, "Continuous measurement of oxygen saturation level using photoplethysmography signal," in *Biomedical and Pharmaceutical Engineering, 2006. ICBPE 2006. International Conference on.* IEEE, 2006, pp. 504–507.

[69] Y.-s. Yan, C. C. Poon, and Y.-t. Zhang, "Journal of neuroengineering and rehabilitation," *Journal of NeuroEngineering and Rehabilitation*, vol. 2, p. 3, 2005.

[70] R. Libby, "A simple method for reliable footstep detection in embedded sensor platforms," 2009.

.1 APPENDIX A

NAME:

DEVICE NUMBER:

Post-lab Questions

Question 1. If a signal is comprised of only three sinusoidal components, how many peaks are in the frequency-domain representation (Fourier Transform) of this signal?

a. 6 peaks **b.** 2 peaks **c.** No peaks **d.** 3 peaks

Question 2. From the following list, circle all the options that represent a signal.

a. Rain falling on a roof.
b. Jet engine noise.
c. Voltage recorded across a capacitor.
d. Water turbulence.
e. Aircraft in motion.
f. The types of phones in the market.

Question 3. Classify the following signals to be 1-dimensional, 2-dimensional, or 3-dimensional. (Mark the dimensionality as 1D, 2D, or 3D against each option)

a. A photograph of yourself **b.** A video display **c.** An audio recording

Question 4. Most real-world signals such as speech signals, electrocardiograms (ECGs) etc. have a constant frequency over time.

a. True **b.** False

Question 5. The principle behind the photoplethysmogram (PPG) signal is based on:

a. Variability in absorption of different wavelengths of light by the blood.
b. The amount of moisture content in the skin.
c. The adrenalin levels in the body.
d. The amount of contact pressure measured between the sensor and the skin.

Question 6. Which of the following parameters can be monitored using the PPG signal?

a. Heart rate
b. Respiratory Rate
c. Oxygen Saturation in the blood
d. Blood Pressure
e. All of the above

Question 7. What does an accelerometer measure?

a. Acceleration caused by the sum of the forces including the force of gravity acting on the accelerometer.

b. The acceleration due to gravity acting on the accelerometer.

c. The direction of force of gravity acting on the accelerometer.

d. The velocity of movement of the accelerometer.

Question 8. From the following list, circle the possible applications using an accelerometer.

a. Gesture recognition

b. Detect the activity of a person (Ex: Walk, Run, Sit, Sleep etc.)

c. Lie detection

d. Pedometer

Question 9. The characteristic wave complex associated with ECG signals comprise of the Q-wave, R-wave, and S-wave in which of the following orders?

a. QRQS **b.** SRQ **c.** QRS **d.** Can vary for every normal human heart.

Question 10. The following is observed about the Heart Rate Variability (HRV) of a normal and healthy person:

a. It increases with age.

b. It remains constant over measurements obtained during different times of the day.

c. It increases when measured during a period of intense physical activity followed by no activity.

d. Both b and c

Question 11. If the interval between two successive peaks of the ECG is 200 samples long, and the sampling rate is 250 Hz, what is the heart rate per minute?

a. 75 **b.** 72 **c.** 60 **d.** 80

Question 12. The abnormality condition called *Atrial Flutter (AF)* and *Atrial Fibrillation (AFib)* show irregularities predominantly in which of the following wave segments of the ECG signal?

a. Q-wave **b.** T-wave **c.** R-wave **d.** S-wave **e.** P-wave

Question 13. If the amount of oxygen in the blood is high, the oxygen saturation (SpO_2):

a. Decreases **b.** Increases **c.** Remains constant

Question 14. How long did it take to get used to the simulation environment on AJDSP?

a. More than 5 minutes, but less than 10 minutes.

b. Less than five minutes.

c. Less than half an hour.

d. More than half an hour.

Question 15. Write a short note on what you learnt from the ECG exercise about ECG signals.

Question 16. Write a short note on what you learnt by performing the speech processing simulation using AJDSP.

Question 17. Write a short note on what you learnt during the accelerometer data collection exercise with respect to the accelerometer as a sensor, the signal acquired, and its applications.

Question 18. Write a short note on what you learnt during the photoplethysmogram heart rate estimation exercise with respect to the signal acquired, and its applications. Also include a few lines on non-invasive mobile health monitoring

Question 19. What did you like most about the signal processing lab and the AJDSP app? Having been given a brief overview into the field of signal processing, would you consider taking signal processing courses in the future?

Question 20. What did you like most about the signal processing lab and the AJDSP app?

Question 21. Did the AJDSP app crash? If yes, please describe what exercise you were working on and how it happened?

Question 22. What did you like least about the signal processing lab and the AJDSP app?

Question 23. Please provide any other comments or feedback if you wish.

Figure 1: Post-lab questions for the FSE 100 undergraduate workshop.

NAME:

DEVICE NUMBER:

Post-lab Questions

Question 1. Most real-world signals such as speech signals, accelerometer signals, electrocardiograms (ECGs) etc. have a constant frequency over time.

a. True **b.** False

Question 2. The principle behind the photoplethysmogram (PPG) signal is based on:

a. Variability in absorption of different wavelengths of light by the blood.

b. The amount of moisture content in the skin.

c. The adrenalin levels in the body.

d. The amount of contact pressure measured between the sensor and the skin.

Question 3. Which of the following parameters can be monitored using the PPG signal?

a. Heart rate

b. Respiratory Rate

c. Oxygen Saturation in the blood

d. Blood Pressure

e. All of the above

Question 4. What does an accelerometer measure?

a. Instantaneous acceleration due to all the forces acting on the accelerometer.

b. Instantaneous acceleration due to gravity acting on the accelerometer.

c. Instantaneous direction of force of gravity acting on the accelerometer.

d. Instantaneous velocity of movement of the accelerometer.

Question 5. From the following list, circle the possible applications of accelerometers.

a. Gesture recognition

b. Detect the activity of a person (Eg: Walk, Run, Sit, Sleep etc.)

c. Lie detection

d. Pedometer

Question 6. The characteristic wave complex associated with ECG signals comprise of the Q-wave, R-wave, and S-wave in which of the following orders?

a. QRQS **b.** SRQ **c.** QRS **d.** Can vary for every normal human heart.

Question 7. In a normal cardiac cycle, the *atrioventricular (AV)* conduction ratio is:

a. 2:1 **b.** 2:2 **c.** 1:1

Question 8. Given that muscle cells have a negative membrane potential when the heart is at rest, the characteristic ECG wave complex is generated by the process of:

a. *arterial repolarization* **c.** *arterial depolarization*

b. *ventricular depolarization* **d.** *ventricular repolarization*

Question 9. The following is observed about the Heart Rate Variability (HRV) of a normal and healthy person:

a. It increases with age.

b. It remains constant over measurements obtained during different times of the day.

c. It increases when measured during a period of intense physical activity followed by no activity.

d. Both b and c

Question 10. If the interval between two successive peaks of the ECG is 200 samples long, and the sampling rate is 250 Hz, what is the heart rate per minute?

a. 75 **b.** 72 **c.** 60 **d.** 80

Question 11. The abnormality condition called *Atrial Flutter (AF)* and *Atrial Fibrillation (AFib)* show irregularities predominantly in which of the following wave segments of the ECG signal?

a. Q-wave **b.** T-wave **c.** R-wave **d.** S-wave **e.** P-wave

Question 12. An ECG signal sampled at 250 Hz contains high frequency noise components, which of the following coefficients must you discard to reduce the noise effects?

a. Approximation/Scaling coefficients **b.** Detail/Wavelet coefficients

Question 13. When the scale of the wavelet increases, the wavelet width:

a. Increases **b.** Decreases **c.** Is unaffected

Question 14. If the amount of oxygen in the blood is high, the oxygen saturation (SpO_2):

a. Decreases **b.** Increases **c.** Remains constant

Question 15. In an ECG signal, what does the R-R interval correspond to?

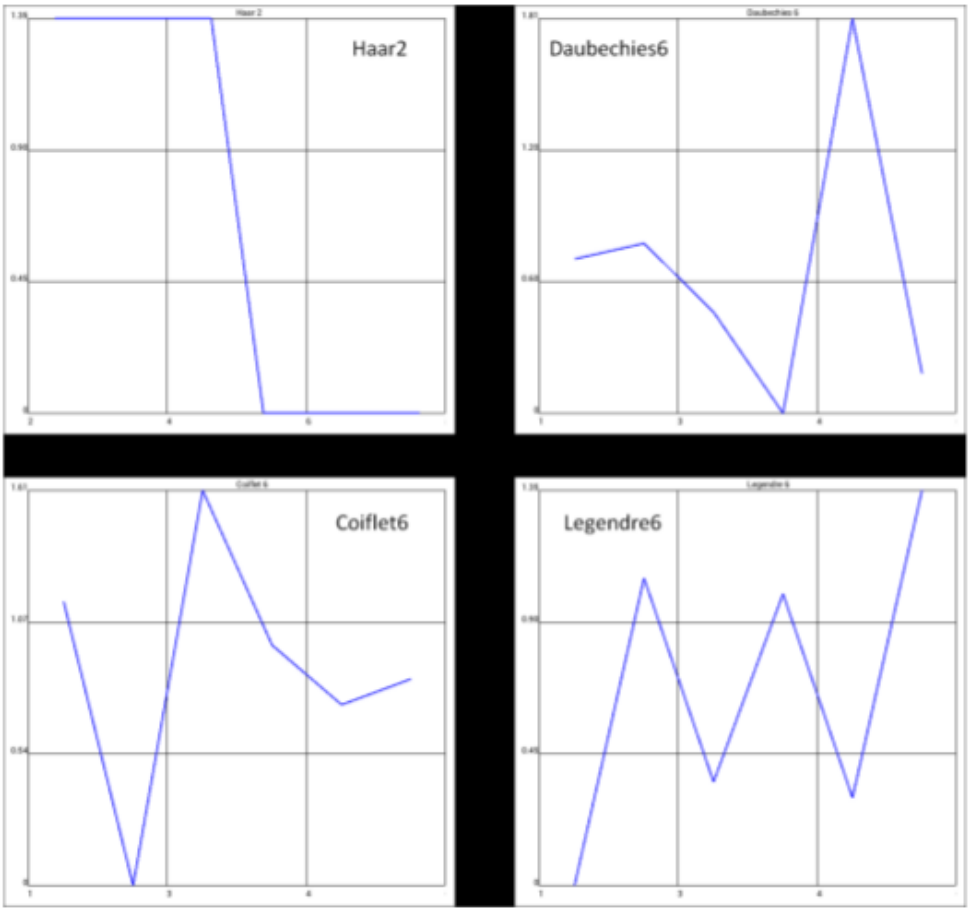
a. Time interval between successive ECG measurements.

b. Time interval between successive heart beats.

c. Time interval between the start and end of a cardiac cycle.

Question 16. Pick the family of wavelets that is most suited for applications involving ECG.

a. Haar **b.** Daubechies6 **c.** Coiflet6 **d.** Legendre6



Question 17. Wavelet transforms can be used in which of the following applications?

a. Signal compression **c.** Edge/Peak detection

b. Noise removal **d.** All of the above

Question 18. How long did it take to get used to the simulation environment using AJDSP?

a. Between 5 – 10 minutes
b. Less than half an hour
c. Less than 5 minutes
d. More than half hour

Question 19. Write a short note on what you learnt from the ECG exercise about ECG signals.

Question 20. Write a short note on what you learnt from the ECG denoising exercise using discrete wavelet transforms (DWT).

Question 21. Write a short note on what you learnt during the accelerometer data collection exercise with respect to the accelerometer as a sensor, the signal acquired, and its applications.

Question 22. Write a short note on what you learnt during the photoplethysmogram heart rate estimation exercise with respect to the signal acquired, and its applications. Also include a few lines on non-invasive mobile health monitoring.

Question 23. What did you like the most about the signal processing lab using AJDSP?

Question 24. What did you like the least about the lab?

Question 25. Did the AJDSP app crash? If yes, please describe what exercise you were working on and how it happened.

Question 26. Please provide additional comments and feedback about AJDSP below.

Figure 2: Post-lab questions for graduate DSP workshop.

NAME:

DEVICE NUMBER:

DSP EXERCISE

Objectives

To provide hands-on experiences with AJDSP, a DSP simulation application built for the Android platform. The exercise problems are directed towards providing a basic understanding of signals and the role of DSP in mobile health monitoring. Signals such as Accelerometer signal, Photoplethysmogram (PPG), and Electrocardiogram (ECG) are introduced. Activities such as observation of signal characteristics, estimation of physiological parameters, and filtering are carried out.

Part 1: Overview of ECG signals

In this part, normal and abnormal ECG signals are visualized using the *Biosignal Generator* block. A brief overview of the various signal characteristics in relation to bio-physical conditions are provided.

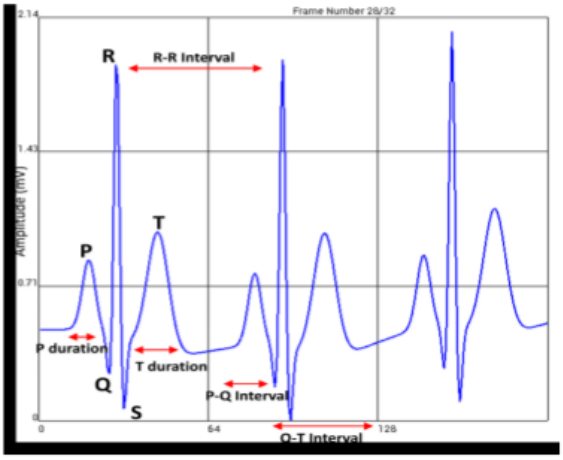


Fig. 1. Characteristic wave segments of an ECG signal.

1. An ECG is a signal obtained by placing electrodes on the surface of the skin to observe the electrical activity of the heart over a period of time.
2. Fig. 1.1 (a) shows a sample ECG recording with its several wave segments annotated.
3. Select the *Biosignal Generator* block from the part list, and choose the 'On-board' option.
4. Set the 'Input Signal' to **Synthetic ECG.**

5. Carefully observe this artificially simulated ECG record in correspondence with the signal in the above figure and identify the P-wave, the QRS-complex and the T-wave which together comprise an ECG waveform for a single heart beat.

- **P-wave** corresponds to wave of **depolarization** starting in the **atria.**
- **QRS-complex** corresponds to electrical excitation of the two **ventricles** and their **depolarization**.
- **T-wave** corresponds to **repolarization of ventricles**, which is the recovery stage of the ventricles restoring them to the rest state.

6. Each QRS-complex with the R-peak corresponds to one beat of the heart.
7. The time between two **R-peaks** is called the **R-R interval**, and it is considered to be the time taken to produce a single heart beat. Heart rate is typically calculated using:

$$\boldsymbol{HBR = 60/R - R\ interval}$$

8. The difference between the successive R-R intervals must be more or less constant in a normal ECG recording. The rate of change of this difference is called **Heart Rate Variability (HRV)**.
9. To navigate through the different frames of the signal, do a swipe action over the plot or use the forward (>>) and backward (<<) frame buttons. Hit the '>' button to automatically scroll through the frames and '||' to pause.
10. Set the 'Input Signal' to **Normal Sinus Rhythm**, this corresponds to a normal clinical ECG recording.
11. Observe the various wave segments discussed above.
12. Set signal to **AFB – Atrial Fibrillation** and notice the P-wave segment.
13. Observe how the QRS-complex and the T-wave follow a regular pattern, while the P-wave is either lost or multiple waves occur.
14. These will in-turn cause a loss in the **Atrioventricular (AV)** conduction ratio, which is the ratio of number of P-waves to the QRS-complexes. A normal ECG waveform will have an AV ratio as 1:1
15. Set signal to **VT – Ventricular Tachycardia** and pay close attention to the T-wave from frames 11 through 13 and frames 27 and 28.
16. Observe the deviation of the T-wave segment from the ideal case.

Part 2: Denoising ECG using Wavelets

In this exercise, wavelets will be used to denoise an ECG signal. The number of scales the signal must be decomposed into will be estimated.

1. In the *Biosignal Generator* block from Part 1, use the pull down menu to set the input signal to **Lead VI w/noise**.

2. Use the frame buttons (>>) or do a swipe action on the plot to navigate to the **2nd frame** of the signal.
3. Tap the Add/Update button to add the block to the workspace.
4. Choose the *Wavelet Transform* block from the list.
5. Set the wavelet type option to **Daubechies6** and tap the 'Add' button to navigate to the workspace.
6. Choose the *Inverse Wavelet Transform* block from the list.
7. Repeat step 5.
8. Choose the *Plot* block from the list and add it to the workspace.
9. Make connections between the blocks and set up the simulation as shown in Fig. 2.

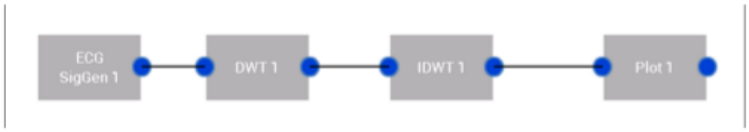


Fig. 2. Simulation to denoise the ECG signal using wavelets.

10. Double tap the *Plot* block to observe the signal. Notice the high frequency noise along the floor of the signal. Does it correspond to the input signal? **Y/N**
11. The simulation above essentially decomposed the input signal into a set of approximation/scaling coefficients (low-frequency) and detail/wavelet coefficients (high-frequency). The signal was then reconstructed using these coefficients and the corresponding wavelet.
12. Now, double tap the DWT block and change the **scale to 3**
13. Notice what happens to the **width of the wavelet** in the plot.
14. Repeat step 12 for the IDWT block as well.
15. Double tap the *Plot* block and observe the plot again. Do you see a difference? **Y/N**
16. What you did was decompose the signal into 3 levels where, in each level the signal was decomposed into two **half-bands of low and high frequency**. Only the low frequency band is further decomposed in every subsequent level into low and high frequency bands.
17. Double tap the DWT block and set the 'Choose Output' option to **Scaling/Low-pass Coeffs**, and observe the output signal in the *Plot* block.
18. By setting the block output to be only the scaling/low-pass coefficients, we have discarded the detail/high-pass coefficients in the input to the IDWT block.
19. Therefore, signal reconstruction was carried out using only a smaller set of coefficients.
20. Repeat step 16. but now setting the 'Choose Output' option to **Wavelet/High-pass Coeffs**.
21. Which among the two signals from step 16 and 17 had a greater amount of noise? _____

22. Based on this observation, set the block output to the option that would allow us to reconstruct a signal without the noise.
23. Change the scale value and estimate the scale that provides the best reconstruction of the input signal with maximum noise elimination.
24. Note down your results below:
    a. Choose Output - _____
    b. Scale - _____
25. Try setting a different wavelet signal (eg: Haar2) in both the DWT and IDWT block, and observe the reconstructed signal.
26. Repeat this for a couple of other wavelets. Which family of wavelets do you think is most suited for applications with ECG signals? _____
27. Press the 'Del All' button to delete all the blocks.

Part 3: Accelerometer and Step Counter

a. Understanding the Accelerometer

1. Choose the *Accelerometer* block from the list, and tap the 'on-board' option.
2. Hold the mobile device with both hands letting it rest with its screen facing the ceiling.
3. Hit the 'start' button and observe the streaming accelerometer signal.
4. Now move the device horizontally along the plane with your hands performing a **left-right** movement. Be careful to keep the device as still as you can in the oriented plane.

Axis	**Movement of the Hand**
X-axis	
Y-axis	
Z-axis	

5. Make a note of which of the three axes signals shows a greater change in its acceleration.
6. Repeat the procedure, now by moving the mobile device **up-down** and observe the signals.

7. Now move the device **forward-back** and again, observe the signals.
8. Use the table above to note down which of the axis corresponds to which of the movements.
9. This should give you an idea of the X, Y, and Z-axis of the mobile device.

b. Data Acquisition and Step Counting

1. Place the mobile device inside your trouser pocket or hold it at the right-side of your waist.
2. You must record a walking movement. To do so, tap the 'start' button and take 5-10 steps. Tap the 'stop' button once you are done walking.
3. Make a note of the number of steps you took; this will be the ground truth data.
4. Make a note of the step count reading on the device.
    - Steps taken (ground truth): _____
    - Device reading: _____
5. Tap the 'save' button; enter the initials of your name followed by the movement. For example, 'DRWalk5steps', and then tap 'OK'. The data is now saved!
    - File name: _____
6. Now scroll through the waveforms by performing a swipe action on the plot and observe the patterns in the signal. View the different signals by selecting different options using the 'Select Signal' pull down menu.
7. Repeat the experiment above by walking a different number of steps between 5 and 10, this time with a slight limp in the gait.
8. Make a note of the steps and save the data with a file name similar to the previous case but, with the word LIMP followed by the initials of your name. (Ex: 'DRLimp7steps')
    - Steps taken (ground truth): _____
    - Device reading: _____
    - File name: _____
9. Repeat step 6.
10. Now perform a sit-stand combo at any pace for any number between 3 to 6 times. Make a note of the number of times you performed this action, and save the data. (Ex. 'DRSitStand4steps').
    - Number of sit-stand actions (ground truth): _____
    - File name: _____
11. Now record running movement and save the data accordingly. (Ex. 'DRRun').
    - File name: _____

Part 4: Photoplethysmogram (PPG)

a. Estimation of Heart Rate and Oxygen Saturation (SpO2)

1. From the list of modules, choose the *PPG* block. Once the camera preview is loaded, place your **left index finger** on the camera; ensure the camera preview shows a display of your finger tip.
2. Tap the 'Start' button to record a video with **normal breathing**.
3. Wait until the timer counts down to zero. Then, remove the finger from the lens.
4. Do a swipe action on the gauge to view the signals and estimated parameters.
5. The signal obtained is called the photoplethysmogram (PPG) also known as blood volume pulse.
6. The peaks in the signal correspond to the beats of your heart.
7. From the 'Select Signal' pull down menu, choose the **Filter Signal** option.
8. Now, analyze the plot and count the number of peaks. Write down the value in the **Manual reading** column of the table provided. Make a note of the parameter values for heart rate, SpO2, and Blue DC value for every experiment in the table.

b. Save the data

9. The following is the naming convention to save the files:
    a. Normal Breathing 1 – 'initialsN1'
    b. Fast Breath – 'initialsFB'
    c. Deep Breath – 'initialsDB'
    d. Normal Breathing 2 – 'initialsN2'
10. Now tap the 'Save' button and enter the corresponding file name and hit 'OK'.
11. **Experiment 2 –** Repeat the procedure for the recording, but with **fast breathing** actions.
12. **Experiment 3** – Repeat the procedure for the recording, but with **deep breathing** actions.
13. **Experiment 4 -** Repeat the procedure for the recording, but with **normal breathing** actions, this time pay close attention to how many times you breathe during the 20 seconds and note it down in the ground truth table.
14. Wait for the organizers to come and help you record the ground truth using the pulse oximeter for each experiment. Do you observe a relationship between the breathing and the SpO_2 values?

Reading from the PPG module

Breath Type	Heart Rate	Oxygen Saturation(SpO2)	PPG Peaks	File Name	Sampling Rate

Ground Truth from Pulse Oximeter

Breath Type	Heart Rate	Oxygen Saturation	Breathing Rate

Part 5: Speech Signal Processing

The primary objective of this exercise is to design a **band-stop** filter to remove a noise signal from the input speech signal using the frame-by-frame processing feature of AJDSP.

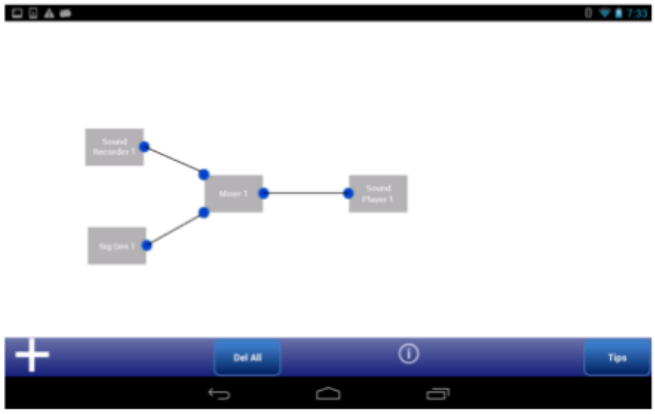


Fig. 2.1

1. Select the *Sound Recorder* block from the part list and tap the 'Add' button.

2. To add a noise signal choose the *Signal Generator* and set signal type as **Sinusoid** with a pulsewidth of **256 samples**, gain **0.04** and frequency **0.6 π**, and press the 'Add' button.
3. Next, choose the *Adder* block followed by the *Sound Player* block and tap 'Add' to add the two blocks to the workspace.
4. Make connections between them to set up the simulation as shown in figure 2.1.
5. Double tap the *Sound Recorder* block to open a view and press the 'Record' button to record your voice up to 10 seconds using the device microphone.
6. Use the 'Stop' button to stop the recording.
7. Press 'Parse' to transfer the data and 'Update' to confirm the change. Frame Size is set to the default **256 samples**.
8. Now double tap on the *Sound Player* block to listen to the noisy signal. (**Ensure that the device is not set on mute**).

a. Denoising using Band-stop Filter

1. Delete the connection between the *Adder* and the *Sound Player* block.
2. Select the *Junction, Filter*, *FIR Design*, and *Frequency Response* blocks from the part list and add them to the workspace.
3. Set up another simulation as shown in figure 2.2. using the above added blocks.

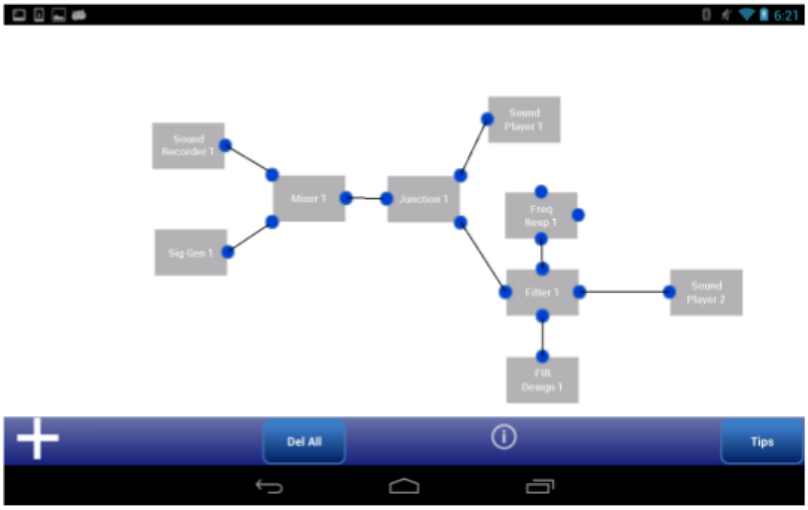


Fig. 2.2

4. Double tap the *FIR Design* block and design a Band-Stop (notch) filter to remove the noisy tone using the following specifications:

Window	Order	Filter-Type	W_1	W_2
Hanning	**32**	**Band-Stop**	**0.5π**	**0.7π**

5. Press 'Add' and 'Done' in the *FIR Design* menu.
6. To process the signal using the filter, double tap the *Sound Recorder* block and press the 'Parse' button followed by the 'Update' button to go back to the workspace.
7. Double tap the *Frequency Response* block and select 'Magnitude' to observe the notch at the frequency of the noisy tone.
8. Double tap the player block and listen to the filtered signal.

b. Fast Fourier Transform

1. To observe the signals in the frequency domain, we need to compute the Fourier Transform of the signal.
2. To do so, delete both the *Sound Player* blocks and replace them with an *FFT* and a *Plot* block as shown in figure 2.3.
3. Double tap *Plot 1* block and select the 'Magnitude' option to observe the FFT of the noise sinusoid. Observe the amplitude of this FFT peak.
4. Similarly, double tap *Plot 2* block and select 'Magnitude' and observe the amplitude of the sinusoid's FFT peak.

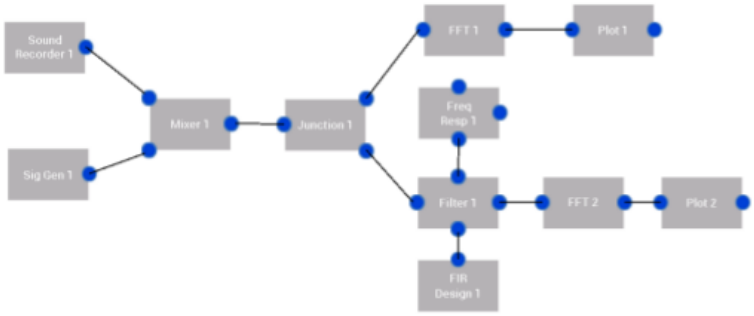


Fig. 2.3

Do the amplitudes of the FFT peaks change after the de-noising operation? (Y/N)_____

How does this correspond to what you heard?

Figure 3: Workshop exercises.